Inside the Research C# Compiler

David R. Hanson and Todd A. Proebsting
Programming Language Systems Group
Microsoft Research
http://www.research.microsoft.com/pls/

Motivation

- C# presents an opportunity for new language features to see wide use in real applications
- Language design research is experimental—MS programmers don't buy paper designs
- .NET provides much of the required infrastructure
- Missing piece: a C# compiler that is
  - Complete
  - Flexible
  - Easy to understand
  - Easy to modify
  - Reasonably efficient

Results in a Nutshell

- Research C# Compiler aka lcsc (local Csharp compiler)
- Written in C#; grok in 1-2 weeks
- Stats:

6735 lines	lexer, parser, data structures (required)
6948	traditional semantic passes (required for "normal" compiler)
5299	optional utility passes
526	program generation tools

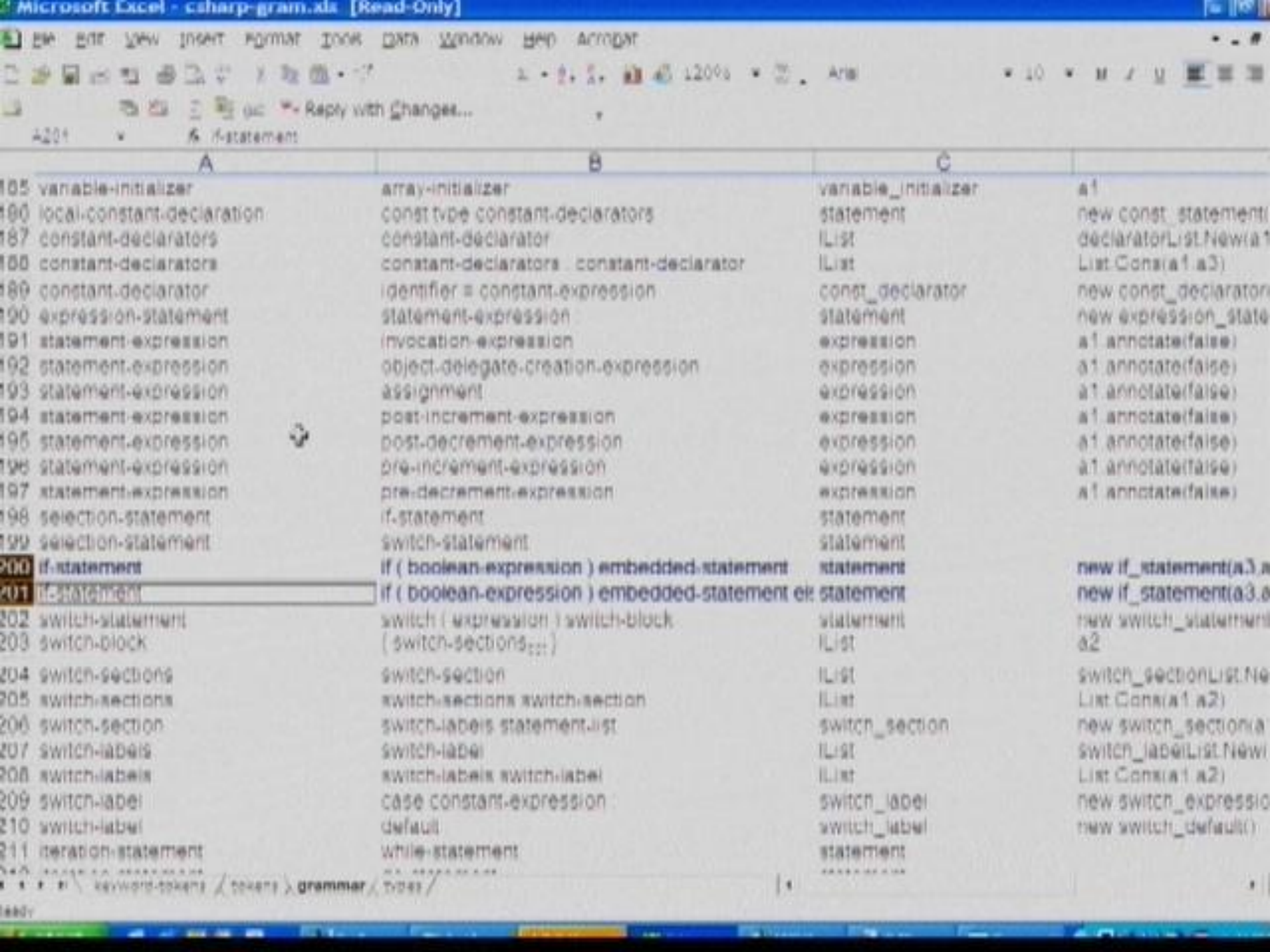
- Simple tools: small 100s of lines
- Complete passes, complex tools: small 1000s of lines
- Typical language extensions: 500-1000 lines; sky's the limit
- Surprise: Most usage to date is for C# tools, not language design

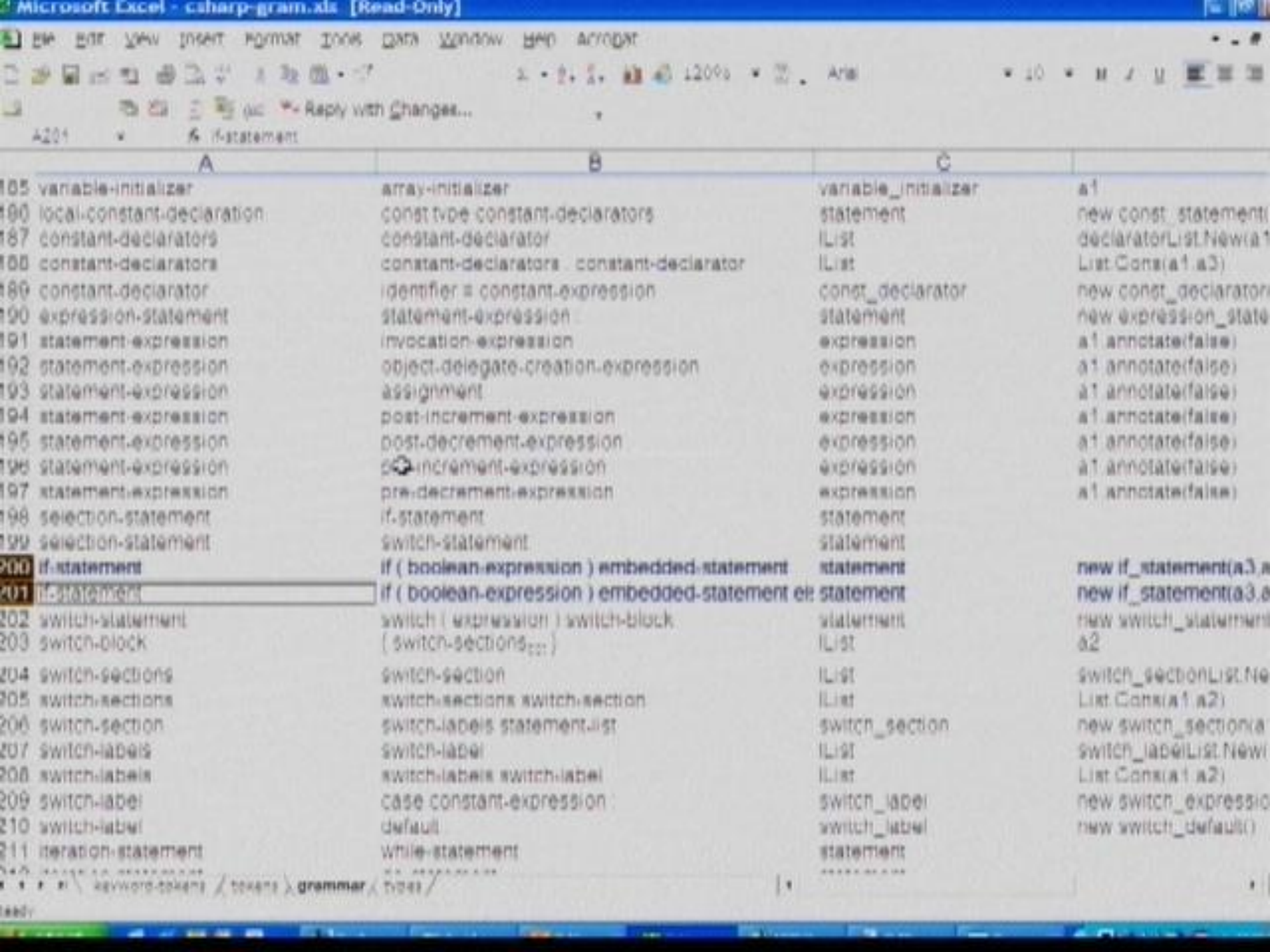
Design From 30,000 Feet

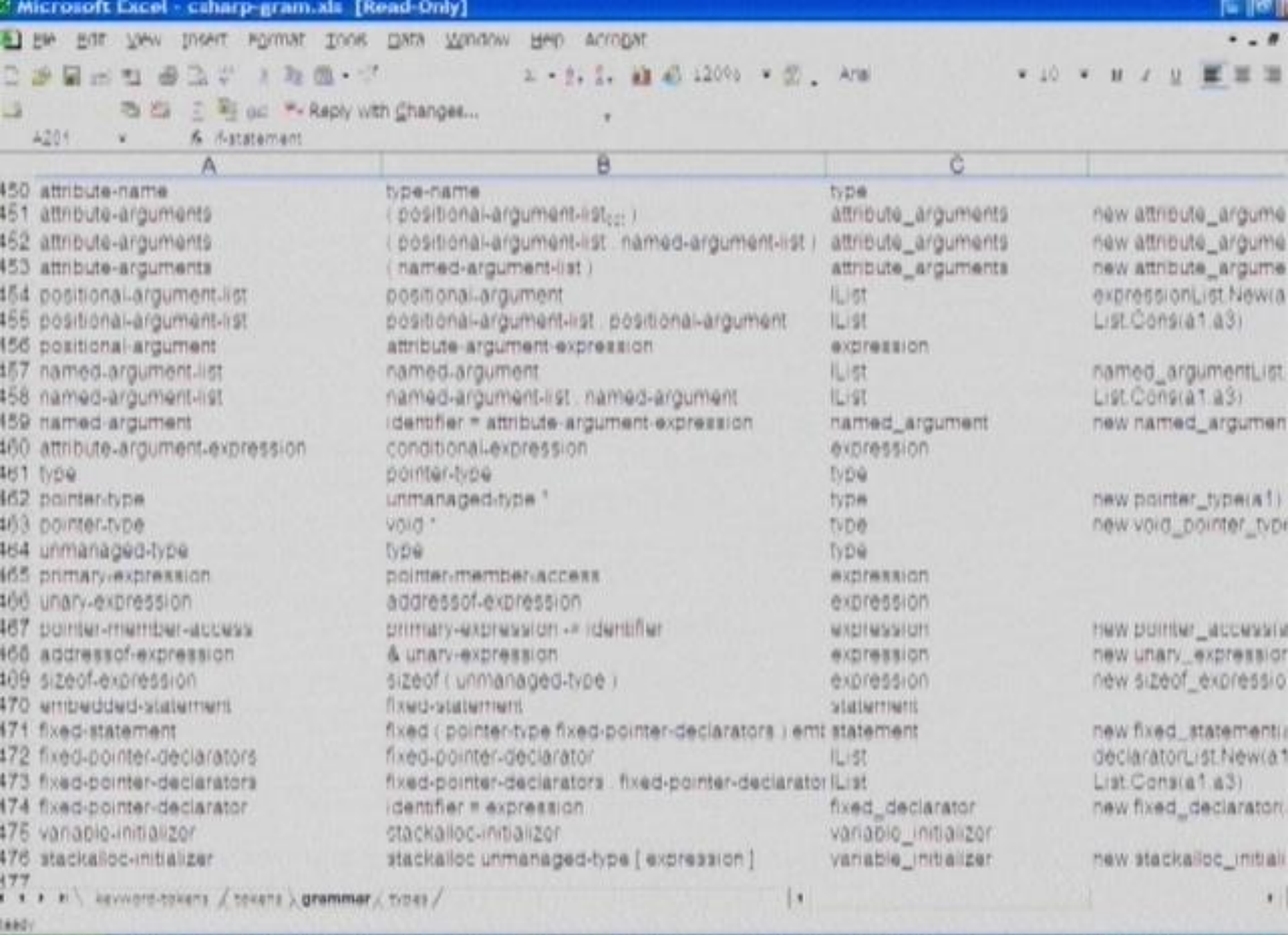
- Dead simple
  - Parser reads C# source, builds an Abstract Syntax Tree (AST)
  - Subsequent passes traverse/modify/annotate AST
- Loosely coupled
  - Passes specified at execution time by command-line options
  - No fixed passes, no fixed order
  - No specified output or output format
- Tool-Driven
  - Parser generator
  - Generator for pass skeletons
  - Strongly typed list generator (poor man's generics)

Parsing: Source to AST

- Generalized LR parser generated automatically from possibly ambiguous C# grammar
- Parser
  - Accepts C# source
  - Builds parse trees (>1 if ambiguous)
  - Builds AST from selected parse tree using semantic actions to build AST nodes
- Grammar
  - Follows published C# grammar closely
  - Specified in Excel spreadsheet [Demo]
  - Excel formulas help get types correct
- Hand-written lexer


Microsoft Excel - csharp-gram.xls [Read-Only]
Reply with Changes...
A201
if-statement
A
B
C
185 variable-initializer
array-initializer
variable_initializer
a1
186 local-constant-declaration
const type constant-declarators
statement
new const_statement(
187 constant-declarators
constant-declarator
IList
declaratorList.New(a1
188 constant-declarators
constant-declarators , constant-declarator
IList
List.Cons(a1,a3)
189 constant-declarator
identifier = constant-expression
const_declarator
new const_declarator(
190 expression-statement
statement-expression
statement
new expression_state
191 statement-expression
invocation-expression
expression
a1.annotate(false)
192 statement-expression
object-delegate-creation-expression
expression
a1.annotate(false)
193 statement-expression
assignment
expression
a1.annotate(false)
194 statement-expression
post-increment-expression
expression
a1.annotate(false)
195 statement-expression
post-decrement-expression
expression
a1.annotate(false)
196 statement-expression
pre-increment-expression
expression
a1.annotate(false)
197 statement-expression
pre-decrement-expression
expression
a1.annotate(false)
198 selection-statement
if-statement
statement
199 selection-statement
switch-statement
statement
200 if-statement
if (boolean-expression) embedded-statement
statement
new if_statement(a3,a
201 if-statement
if (boolean-expression) embedded-statement el
statement
new if_statement(a3,a
202 switch-statement
switch (expression) switch-block
statement
new switch_statement(
203 switch-block
{ switch-sections_opt }
IList
a2
204 switch-sections
switch-section
IList
switch_sectionList.Ne
205 switch-sections
switch-sections switch-section
IList
List.Cons(a1,a2)
206 switch-section
switch-labels statement-list
switch_section
new switch_section(a
207 switch-labels
switch-label
IList
switch_labelList.New(
208 switch-labels
switch-labels switch-label
IList
List.Cons(a1,a2)
209 switch-label
case constant-expression :
switch_label
new switch_expressio
210 switch-label
default :
switch_label
new switch_default()
211 iteration-statement
while-statement
statement
keyword-tokens
tokens
grammar
types
Ready


Microsoft Excel - csharp-gram.xls [Read-Only]
Reply with Changes...
A201
if-statement
A
B
C
185 variable-initializer
array-initializer
variable_initializer
a1
190 local-constant-declaration
const type constant-declarators
statement
new const_statement(
187 constant-declarators
constant-declarator
IList
declaratorList.New(a1
188 constant-declarators
constant-declarators , constant-declarator
IList
List.Cons(a1,a3)
189 constant-declarator
identifier = constant-expression
const_declarator
new const_declarator(
190 expression-statement
statement-expression ;
statement
new expression_state
191 statement-expression
invocation-expression
expression
a1.annotate(false)
192 statement-expression
object-delegate-creation-expression
expression
a1.annotate(false)
193 statement-expression
assignment
expression
a1.annotate(false)
194 statement-expression
post-increment-expression
expression
a1.annotate(false)
195 statement-expression
post-decrement-expression
expression
a1.annotate(false)
196 statement-expression
pre-increment-expression
expression
a1.annotate(false)
197 statement-expression
pre-decrement-expression
expression
a1.annotate(false)
198 selection-statement
if-statement
statement
199 selection-statement
switch-statement
statement
200 if-statement
if (boolean-expression) embedded-statement
statement
new if_statement(a3,a
201 if-statement
if (boolean-expression) embedded-statement el
statement
new if_statement(a3,a
202 switch-statement
switch (expression) switch-block
statement
new switch_statement
203 switch-block
{ switch-sections_opt }
IList
a2
204 switch-sections
switch-section
IList
switch_sectionList.Ne
205 switch-sections
switch-sections switch-section
IList
List.Cons(a1,a2)
206 switch-section
switch-labels statement-list
switch_section
new switch_section(a
207 switch-labels
switch-label
IList
switch_labelList.New(
208 switch-labels
switch-labels switch-label
IList
List.Cons(a1,a2)
209 switch-label
case constant-expression :
switch_label
new switch_expressio
210 switch-label
default :
switch_label
new switch_default()
211 iteration-statement
while-statement
statement
keywords-tokens
tokens
grammar
types
Ready


Microsoft Excel - csharp-gram.xls [Read-Only]
File Edit View Insert Format Tools Data Window Help Acrobat
Reply with Changes...
A201 fixed-statement
A B C
450 attribute-name type-name type
451 attribute-arguments (positional-argument-list(opt)) attribute_arguments new attribute_argume
452 attribute-arguments (positional-argument-list , named-argument-list) attribute_arguments new attribute_argume
453 attribute-arguments (named-argument-list) attribute_arguments new attribute_argume
454 positional-argument-list positional-argument IList expressionList.New(a
455 positional-argument-list positional-argument-list , positional-argument IList List.Cons(a1,a3)
456 positional-argument attribute-argument-expression expression
457 named-argument-list named-argument IList named_argumentList
458 named-argument-list named-argument-list , named-argument IList List.Cons(a1,a3)
459 named-argument identifier = attribute-argument-expression named_argument new named_argumen
460 attribute-argument-expression conditional-expression expression
461 type pointer-type type
462 pointer-type unmanaged-type * type new pointer_type(a1)
463 pointer-type void * type new void_pointer_type
464 unmanaged-type type type
465 primary-expression pointer-member-access expression
466 unary-expression addressof-expression expression
467 pointer-member-access primary-expression -> identifier expression new pointer_accessio
468 addressof-expression & unary-expression expression new unary_expressio
469 sizeof-expression sizeof (unmanaged-type) expression new sizeof_expressio
470 embedded-statement fixed-statement statement
471 fixed-statement fixed (pointer-type fixed-pointer-declarators) emt statement new fixed_statement(
472 fixed-pointer-declarators fixed-pointer-declarator IList declaratorList.New(a1
473 fixed-pointer-declarators fixed-pointer-declarators , fixed-pointer-declarator IList List.Cons(a1,a3)
474 fixed-pointer-declarator identifier = expression fixed_declarator new fixed_declarator
475 variable-initializer stackalloc-initializer variable_initializer
476 stackalloc-initializer stackalloc unmanaged-type [expression] variable_initializer new stackalloc_initial
477
keyword-tokens tokens grammar types
Ready

Parsing: Source to AST

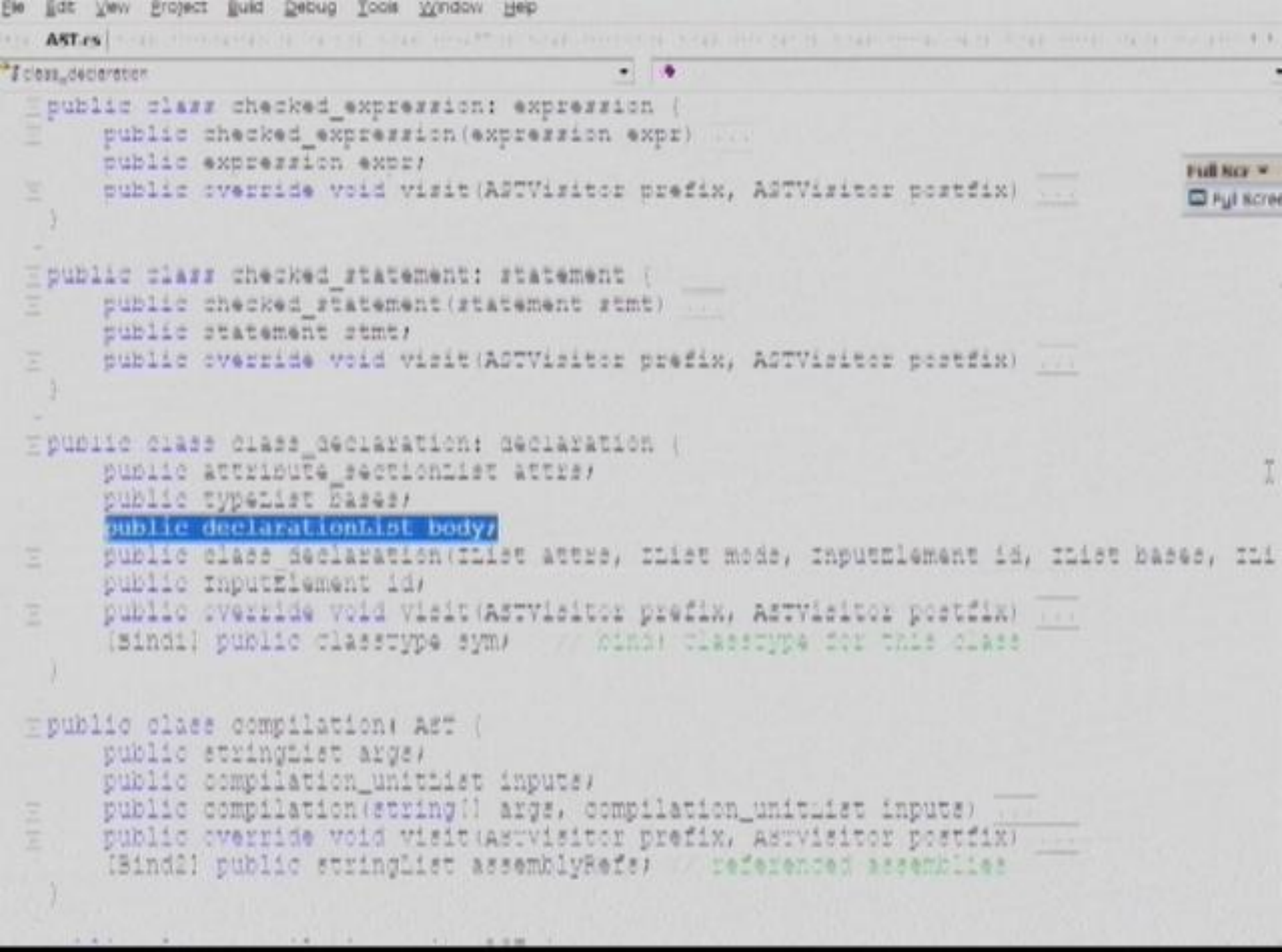
- Generalized LR parser generated automatically from possibly ambiguous C# grammar
- Parser
  - Accepts C# source
  - Builds parse trees (>1 if ambiguous)
  - Builds AST from selected parse tree using semantic actions to build AST nodes
- Grammar
  - Follows published C# grammar closely
  - Specified in Excel spreadsheet [Demo]
  - Excel formulas help get types correct
- Hand-written lexer

Abstract Syntax Trees

- 164 node types; 16 are abstract
- Follows C# syntax closely; strongly typed

```
public class if_statement: statement {
    public if_statement(expression expr,
            statement thenpart, statement elsepart) {
        this.expr = expr;
        this.thenpart = thenpart;
        this.elsepart = elsepart;
    }
    public expression expr;
    public statement thenpart;
    [MayBeNull] public statement elsepart;
    public override void visit(ASTVisitor prefix,
            ASTVisitor postfix) { ... }
}
```

- Lists are strongly typed [Demo]


File Edit View Project Build Debug Tools Window Help
AST.cs
class_decorator
public class checked_expression: expression {
public checked_expression(expression expr)
public expression expr;
public override void visit(ASTVisitor prefix, ASTVisitor postfix)
}
public class checked_statement: statement {
public checked_statement(statement stmt)
public statement stmt;
public override void visit(ASTVisitor prefix, ASTVisitor postfix)
}
public class class_declaration: declaration {
public attribute_sectionList attrs;
public typeList bases;
public declarationList body;
public class_declaration(tList attrs, tList mods, inputElement id, tList bases, tLi
public inputElement id;
public override void visit(ASTVisitor prefix, ASTVisitor postfix)
[bind1] public classType sym; // bind1 classType for this class
}
public class compilation: AST {
public stringList args;
public compilation_unitList inputs;
public compilation(string[] args, compilation_unitList inputs)
public override void visit(ASTVisitor prefix, ASTVisitor postfix)
[bind2] public stringList assemblyRefs; // referenced assemblies
}
Full Scr
Full Scree

Abstract Syntax Trees

- 164 node types; 16 are abstract
- Follows C# syntax closely; strongly typed

```
public class if_statement: statement {
    public if_statement(expression expr,
            statement thenpart, statement elsepart) {
        this.expr = expr;
        this.thenpart = thenpart;
        this.elsepart = elsepart;
    }
    public expression expr;
    public statement thenpart;
    [MayBeNull] public statement elsepart;
    public override void visit(ASTVisitor prefix,
            ASTVisitor postfix) { ... }
}
```

- Lists are strongly typed [Demo]

Semantics

- Semantic pass == AST traversal
- "Built-in" visit method on each AST node: Supply a delegate

```
public class if_statement: statement {
    ...
    public override void visit(ASTVisitor prefix,
            ASTVisitor postfix) {
        prefix(this);
        expr.visit(prefix, postfix);
        thenpart.visit(prefix, postfix);
        if (elsepart != null)
            elsepart.visit(prefix, postfix);
        postfix(this);
    }
}
```

- Complete, pass-specific AST traversal: Write a class with one method per AST node type

Simple C# Tools

- Use visit method for simple grep-like C# tools

```
public static void prefix() { … }
public static void postfix() { … }
public static void Main(string[] args) {
    foreach (string s in args) {
        AST ast = (AST)Parser.parse(s,
                                    new StreamReader(s));
        if (ast != null)
            ast.visit(new ASTVisitor(prefix),
                      new ASTVisitor(postfix));
    }
}
```

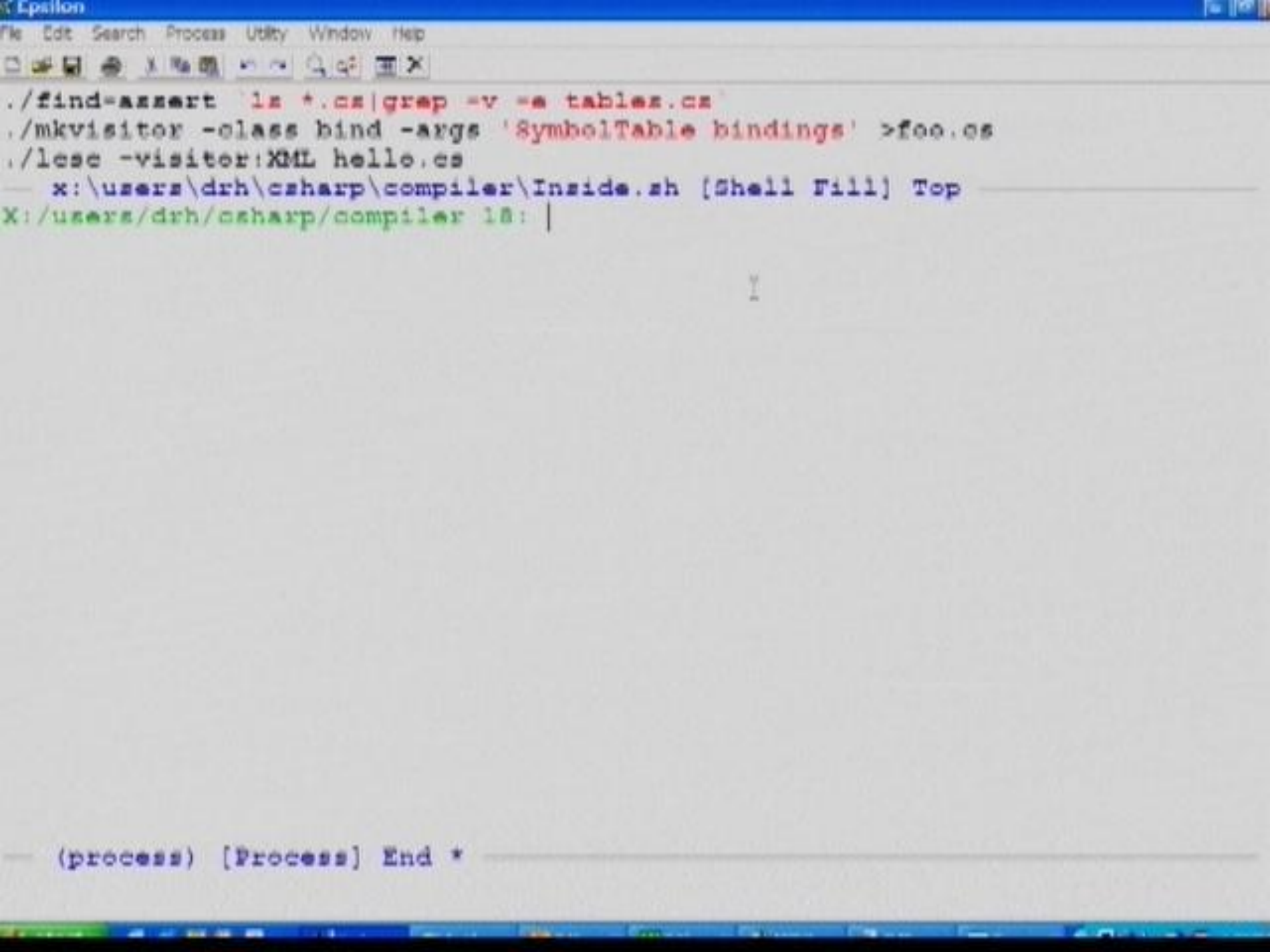
- Unlike grep, AST tools can consider context
- Tools to compute code metrics are easy

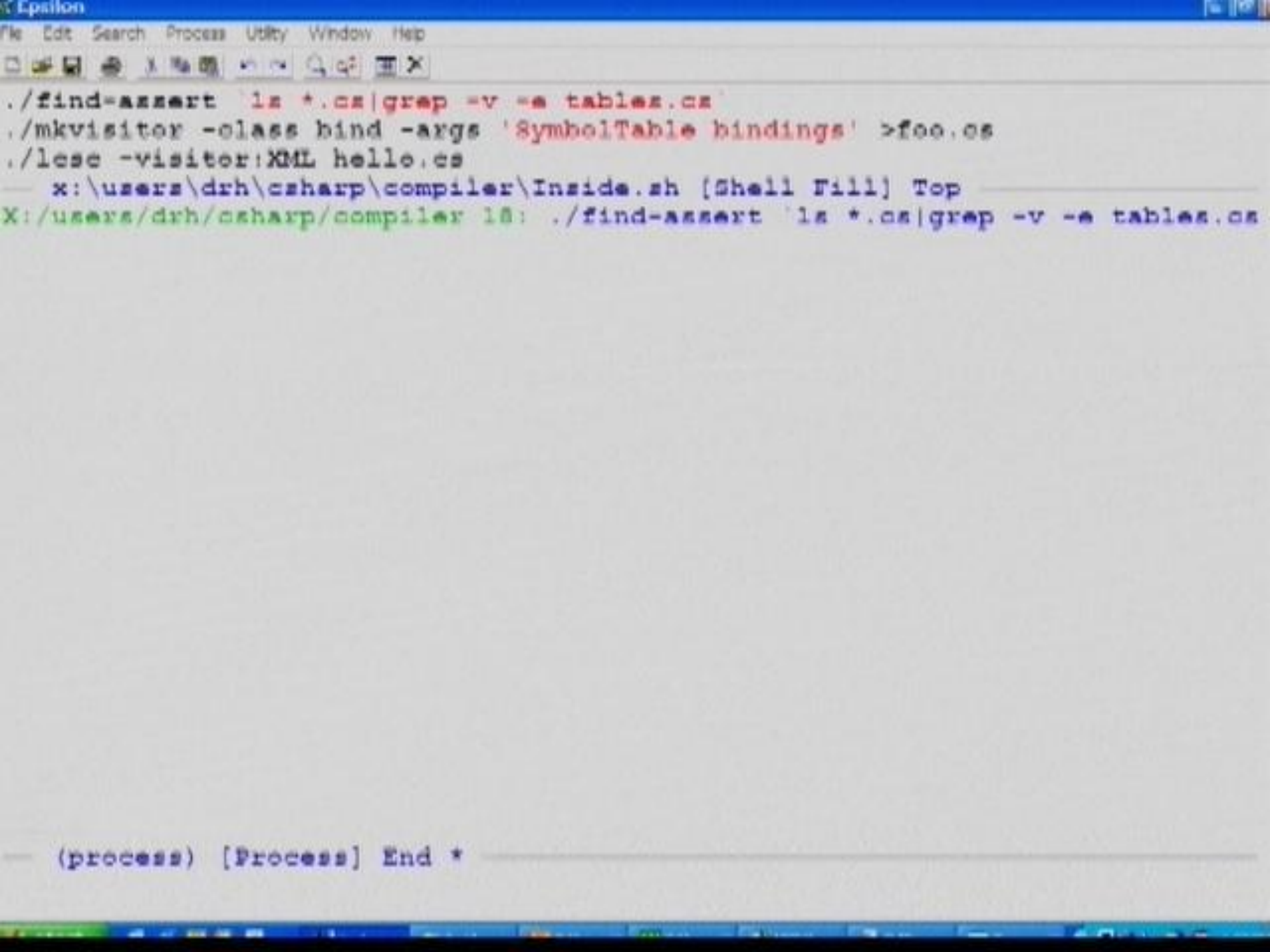
Finding Assertions

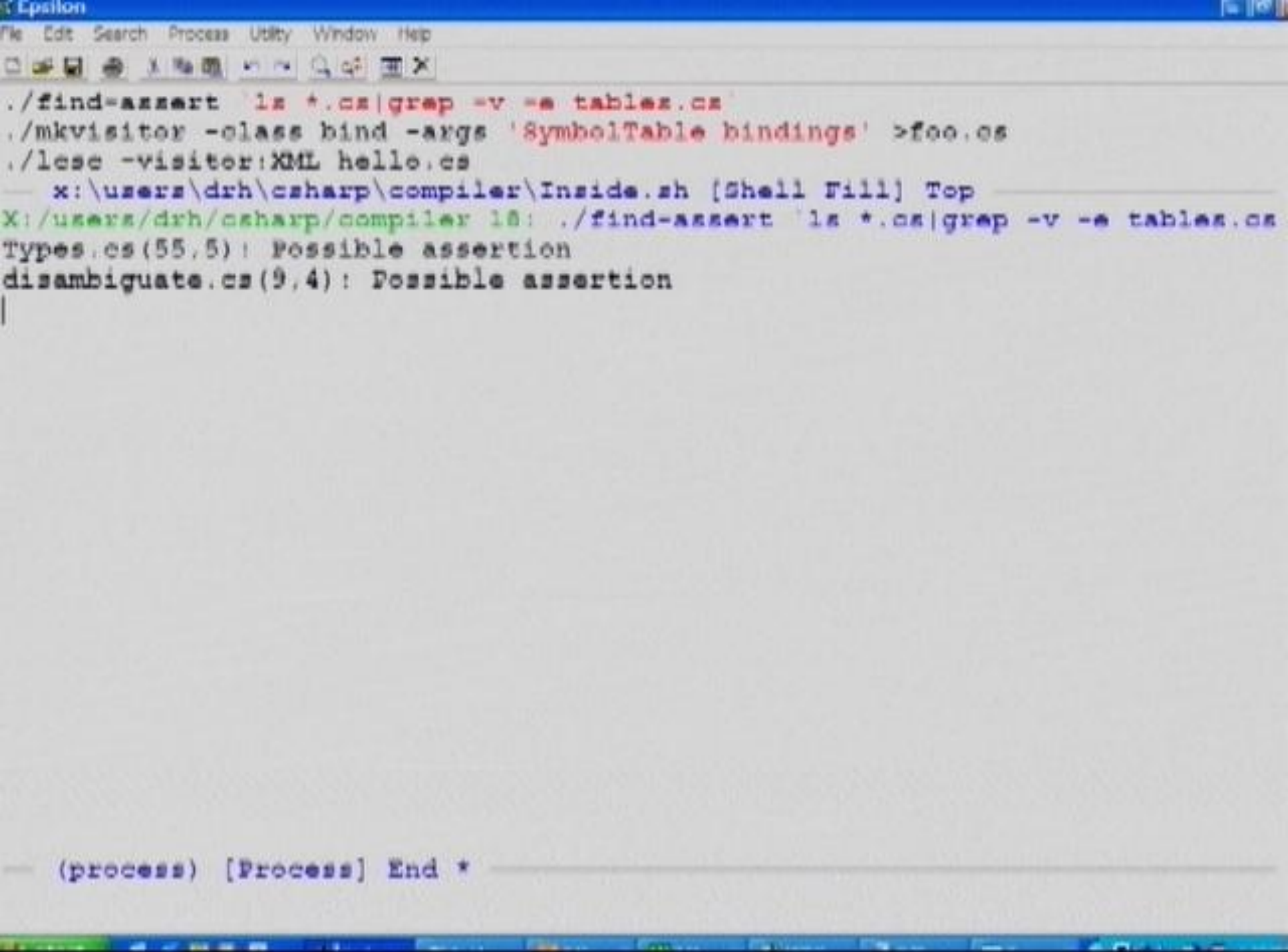
- Find potential assertions:

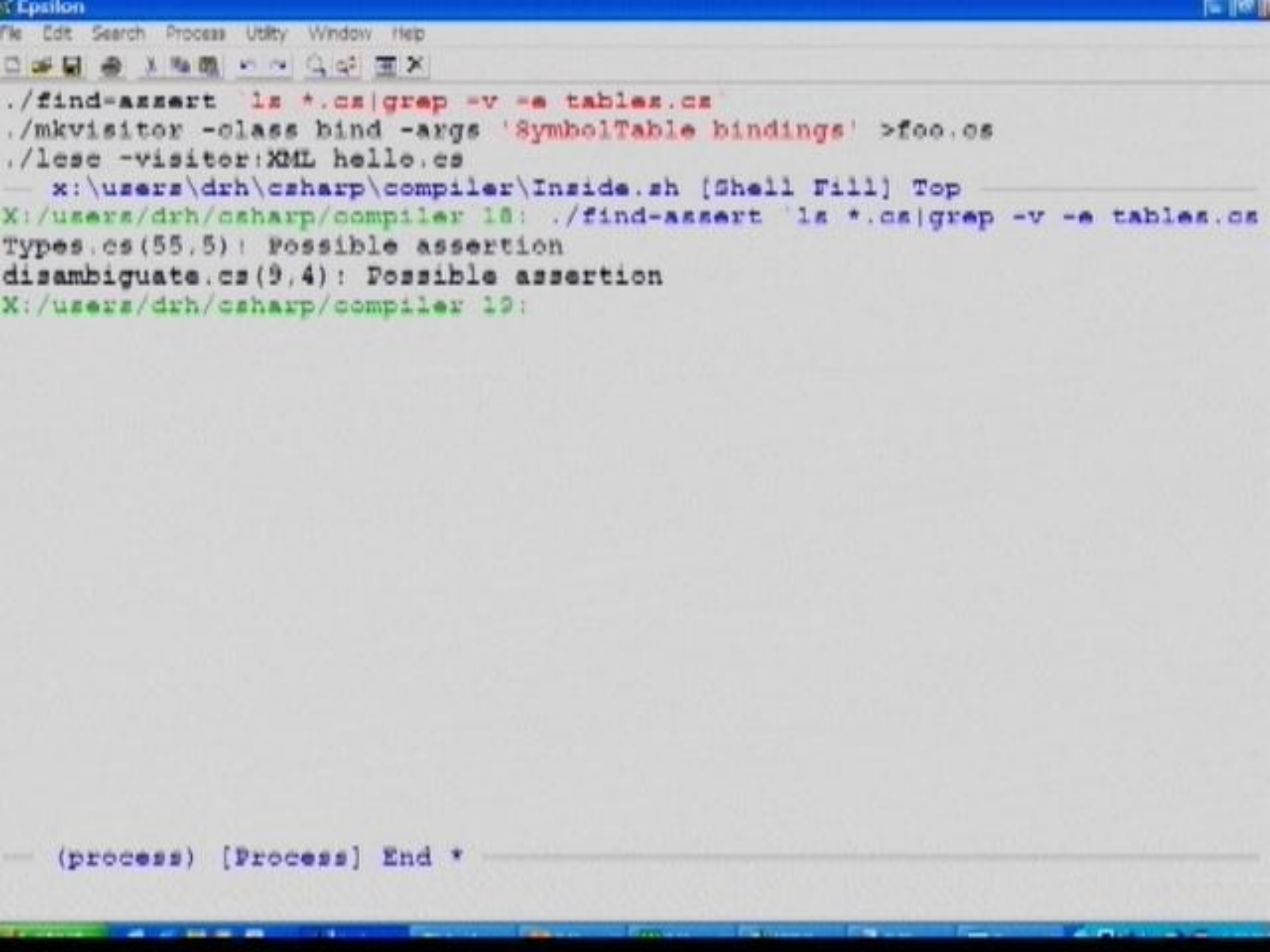
```
if (…) throw …;

public static void doit(AST ast) {
   if (ast is if_statement
   && ((if_statement)ast).thenpart is throw_statement
   && ((if_statement)ast).elsepart == null)
       Console.WriteLine("{0}: Possible assertion",
           ast.begin);
}

public static void Main(string[] args) {
   …
   ast.visit(new ASTVisitor(doit));
}
```

- Thanks to Rustan Leino [Demo]


Epsilon
File Edit Search Process Utility Window Help
./find-assert 'ls *.cs|grep -v -e tables.cs'
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./lese -visitor:XML hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] Top
X:/users/drh/csharp/compiler 18:
(process) [Process] End *


Epsilon
File Edit Search Process Utility Window Help
./find-assert 'ls *.cs|grep -v -e tables.cs'
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./lese -visitor:XML hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] Top
X:/users/drh/csharp/compiler 18: ./find-assert 'ls *.cs|grep -v -e tables.cs
(process) [Process] End *


Epsilon
File Edit Search Process Utility Window Help
./find-assert 'ls *.cs|grep -v -e tables.cs'
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./lcsc -visitor:XML hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] Top
X:/users/drh/csharp/compiler 18: ./find-assert 'ls *.cs|grep -v -e tables.cs
Types.cs(55,5): Possible assertion
disambiguate.cs(9,4): Possible assertion
(process) [Process] End *


Epsilon
File Edit Search Process Utility Window Help
./find-assert 'ls *.cs|grep -v -e tables.cs'
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./lcsc -visitor:XML hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] Top
X:/users/drh/csharp/compiler 18: ./find-assert 'ls *.cs|grep -v -e tables.cs
Types.cs(55,5): Possible assertion
disambiguate.cs(9,4): Possible assertion
X:/users/drh/csharp/compiler 19:
(process) [Process] End *

Finding Assertions

- Find potential assertions:

```
if (…) throw …;

public static void doit(AST ast) {
   if (ast is if_statement
   && ((if_statement)ast).thenpart is throw_statement
   && ((if_statement)ast).elsepart == null)
       Console.WriteLine("{0}: Possible assertion",
           ast.begin);
}

public static void Main(string[] args) {
   …
   ast.visit(new ASTVisitor(doit));
}
```

- Thanks to Rustan Leino [Demo]

Visitors

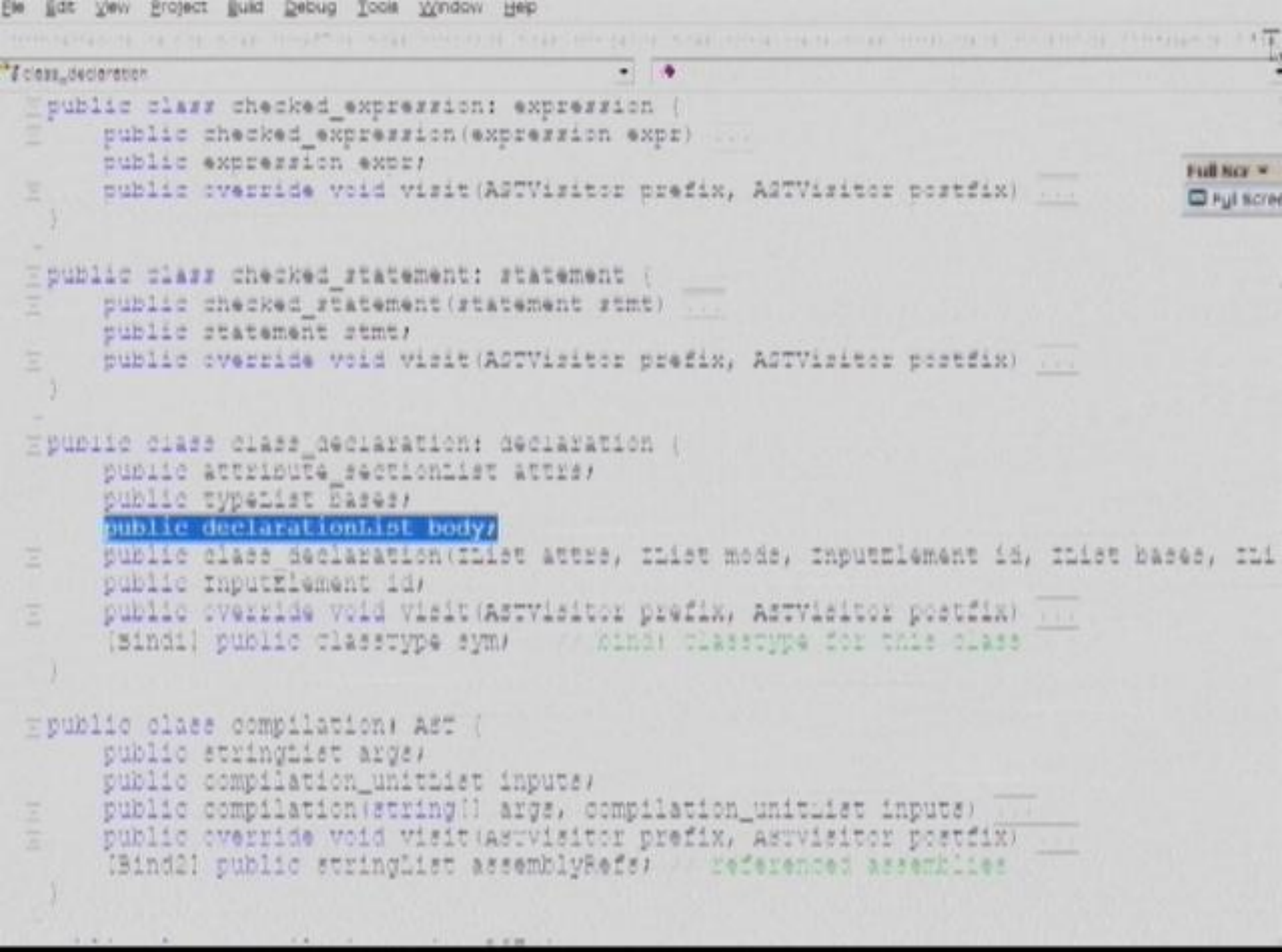
- AST visitor is any class with

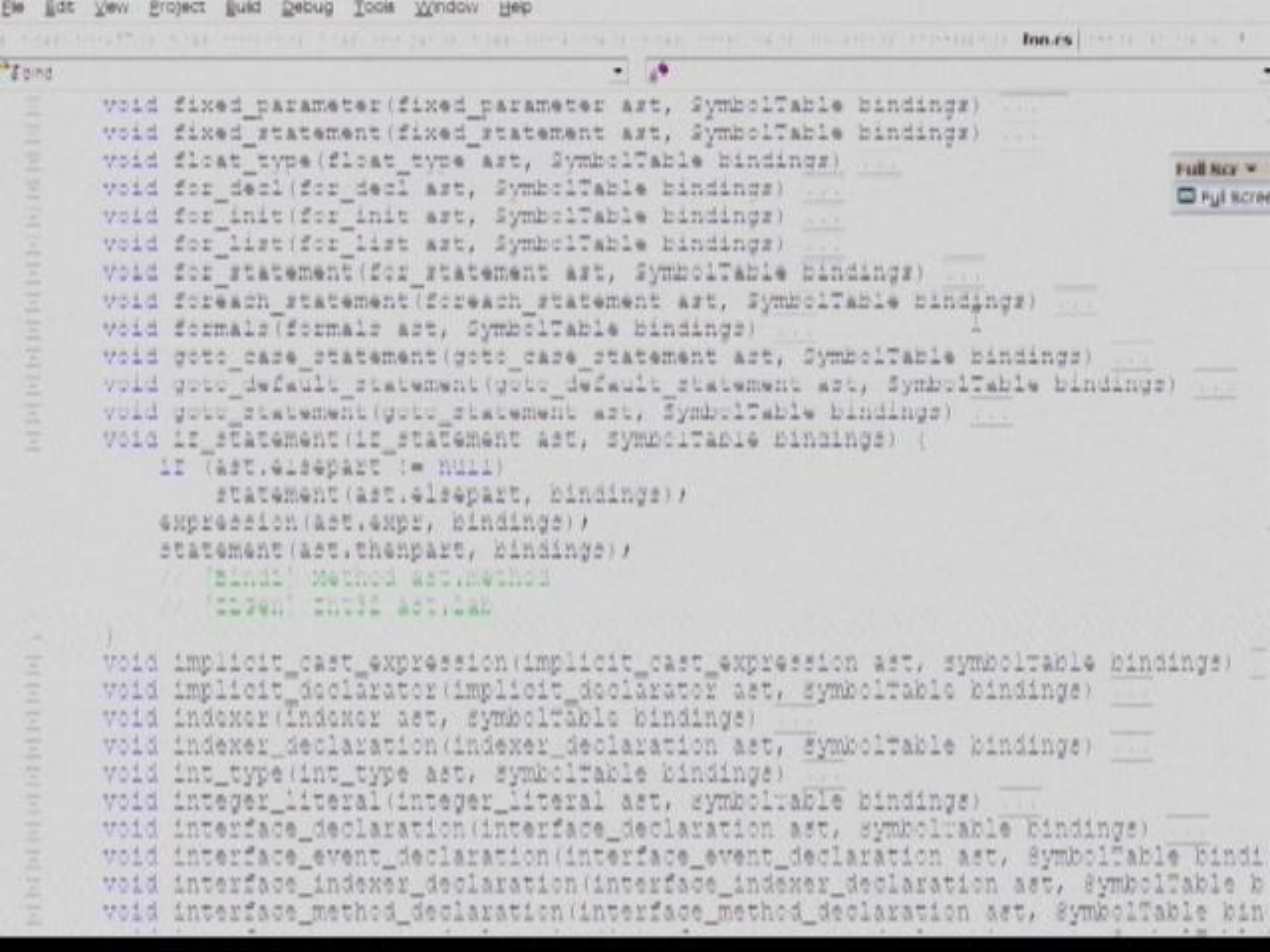
  ```
  public static object visit(object ast, TextWriter w);
  ```

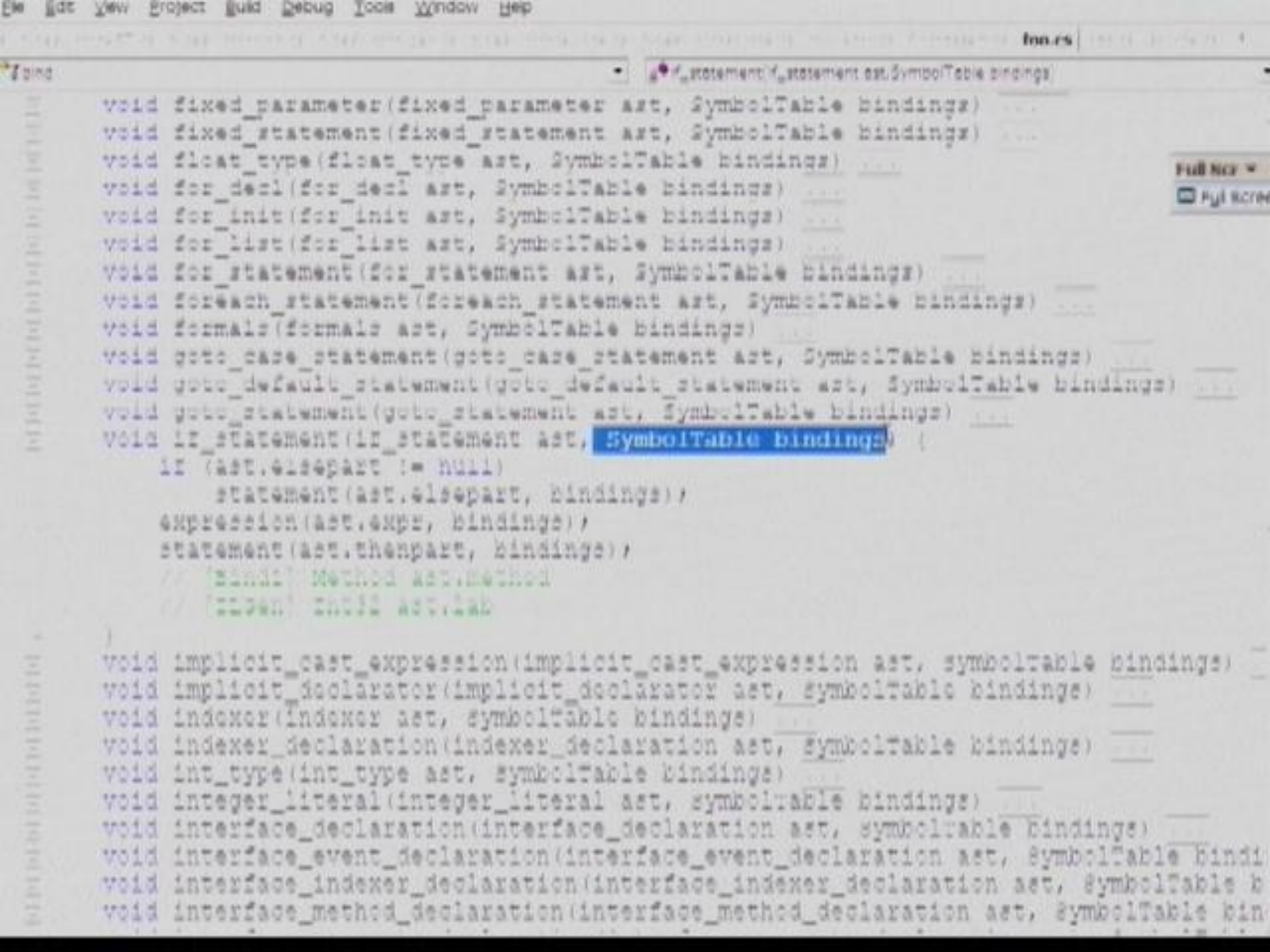
  - Accepts and returns an AST
  - Emit output, if any, to w
- Visitors return the AST, which is passed to others
- Command-line options give visitors, order of invocation

  lcsc -visitor:*classname,file ... C# source files*
  - Omit *file* if *file == classname*
- Visitor class is responsible for traversing the AST
  - mkvisitor generates 800+ lines of skeleton code

    mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
  - [Demo]


File Edit View Project Build Debug Tools Window Help
class_declaration
public class checked_expression: expression {
public checked_expression(expression expr) ...
public expression expr;
public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
}
public class checked_statement: statement {
public checked_statement(statement stmt) ...
public statement stmt;
public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
}
public class class_declaration: declaration {
public attribute_sectionList attrs;
public typeList bases;
public declarationList body;
public class_declaration(IList attrs, IList mods, InputElement id, IList bases, IL
public InputElement id;
public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
[Bind1] public classType sym; // Bind1 classType for this class
}
public class compilation: AST {
public stringList args;
public compilation_unitList inputs;
public compilation(string[] args, compilation_unitList inputs) ...
public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
[Bind2] public stringList assemblyRefs; // referenced assemblies
}
Full Scr
Full scree


File Edit View Project Build Debug Tools Window Help
Bindings
Found
void fixed_parameter(fixed_parameter ast, SymbolTable bindings)
void fixed_statement(fixed_statement ast, SymbolTable bindings)
void float_type(float_type ast, SymbolTable bindings)
void for_decl(for_decl ast, SymbolTable bindings)
void for_init(for_init ast, SymbolTable bindings)
void for_list(for_list ast, SymbolTable bindings)
void for_statement(for_statement ast, SymbolTable bindings)
void foreach_statement(foreach_statement ast, SymbolTable bindings)
void formals(formals ast, SymbolTable bindings)
void goto_case_statement(goto_case_statement ast, SymbolTable bindings)
void goto_default_statement(goto_default_statement ast, SymbolTable bindings)
void goto_statement(goto_statement ast, SymbolTable bindings)
void if_statement(if_statement ast, symboltable bindings) {
if (ast.elsepart != null)
statement(ast.elsepart, bindings);
expression(ast.expr, bindings);
statement(ast.thenpart, bindings);
// Bind! Method ast.method
// Error! Bind! ast.label
}
void implicit_cast_expression(implicit_cast_expression ast, symboltable bindings)
void implicit_declarator(implicit_declarator ast, symboltable bindings)
void indexer(Indexer ast, symboltable bindings)
void indexer_declaration(indexer_declaration ast, symboltable bindings)
void int_type(int_type ast, symboltable bindings)
void integer_literal(integer_literal ast, symboltable bindings)
void interface_declaration(interface_declaration ast, symboltable bindings)
void interface_event_declaration(interface_event_declaration ast, SymbolTable bindi
void interface_indexer_declaration(interface_indexer_declaration ast, SymbolTable b
void interface_method_declaration(interface_method_declaration ast, SymbolTable bin


File Edit View Project Build Debug Tools Window Help
Program.cs
void fixed_parameter(fixed_parameter ast, SymbolTable bindings)
void fixed_statement(fixed_statement ast, SymbolTable bindings)
void float_type(float_type ast, SymbolTable bindings)
void for_decl(for_decl ast, SymbolTable bindings)
void for_init(for_init ast, SymbolTable bindings)
void for_list(for_list ast, SymbolTable bindings)
void for_statement(for_statement ast, SymbolTable bindings)
void foreach_statement(foreach_statement ast, SymbolTable bindings)
void formals(formals ast, SymbolTable bindings)
void goto_case_statement(goto_case_statement ast, SymbolTable bindings)
void goto_default_statement(goto_default_statement ast, SymbolTable bindings)
void goto_statement(goto_statement ast, SymbolTable bindings)
void if_statement(if_statement ast, SymbolTable bindings) {
if (ast.elsepart != null)
statement(ast.elsepart, bindings);
expression(ast.expr, bindings);
statement(ast.thenpart, bindings);
// Build Method ast.method
// Leave Scope ast.Scope
}
void implicit_cast_expression(implicit_cast_expression ast, symboltable bindings)
void implicit_declarator(implicit_declarator ast, symboltable bindings)
void indexer(indexer ast, symboltable bindings)
void indexer_declaration(indexer_declaration ast, symboltable bindings)
void int_type(int_type ast, symboltable bindings)
void integer_literal(integer_literal ast, symboltable bindings)
void interface_declaration(interface_declaration ast, symboltable bindings)
void interface_event_declaration(interface_event_declaration ast, SymbolTable bindi
void interface_indexer_declaration(interface_indexer_declaration ast, SymbolTable b
void interface_method_declaration(interface_method_declaration ast, SymbolTable bin

```
    void fixed_parameter(fixed_parameter ast, SymbolTable bindings)
    void fixed_statement(fixed_statement ast, SymbolTable bindings)
    void float_type(float_type ast, SymbolTable bindings)
    void for_decl(for_decl ast, SymbolTable bindings)
    void for_init(for_init ast, SymbolTable bindings)
    void for_list(for_list ast, SymbolTable bindings)
    void for_statement(for_statement ast, SymbolTable bindings)
    void foreach_statement(foreach_statement ast, SymbolTable bindings)
    void formals(formals ast, SymbolTable bindings)
    void goto_case_statement(goto_case_statement ast, SymbolTable bindings)
    void goto_default_statement(goto_default_statement ast, SymbolTable bindings)
    void goto_statement(goto_statement ast, SymbolTable bindings)
    void if_statement(if_statement ast, symboltable bindings) {
        if (ast.elsepart != null)
            statement(ast.elsepart, bindings);
        expression(ast.expr, bindings);
        statement(ast.thenpart, bindings);
        // Bind: Method ast.method
        // Save: Scope ast.scope
    }
    void implicit_cast_expression(implicit_cast_expression ast, symboltable bindings)
    void implicit_declarator(implicit_declarator ast, symboltable bindings)
    void indexer(indexer ast, symboltable bindings)
    void indexer_declaration(indexer_declaration ast, symboltable bindings)
    void int_type(int_type ast, symboltable bindings)
    void integer_literal(integer_literal ast, symboltable bindings)
    void interface_declaration(interface_declaration ast, symboltable bindings)
    void interface_event_declaration(interface_event_declaration ast, SymbolTable bindi
    void interface_indexer_declaration(interface_indexer_declaration ast, SymbolTable b
    void interface_method_declaration(interface_method_declaration ast, SymbolTable bin
```

Visitors

- AST visitor is any class with

```
public static object visit(object ast, TextWriter w);
```

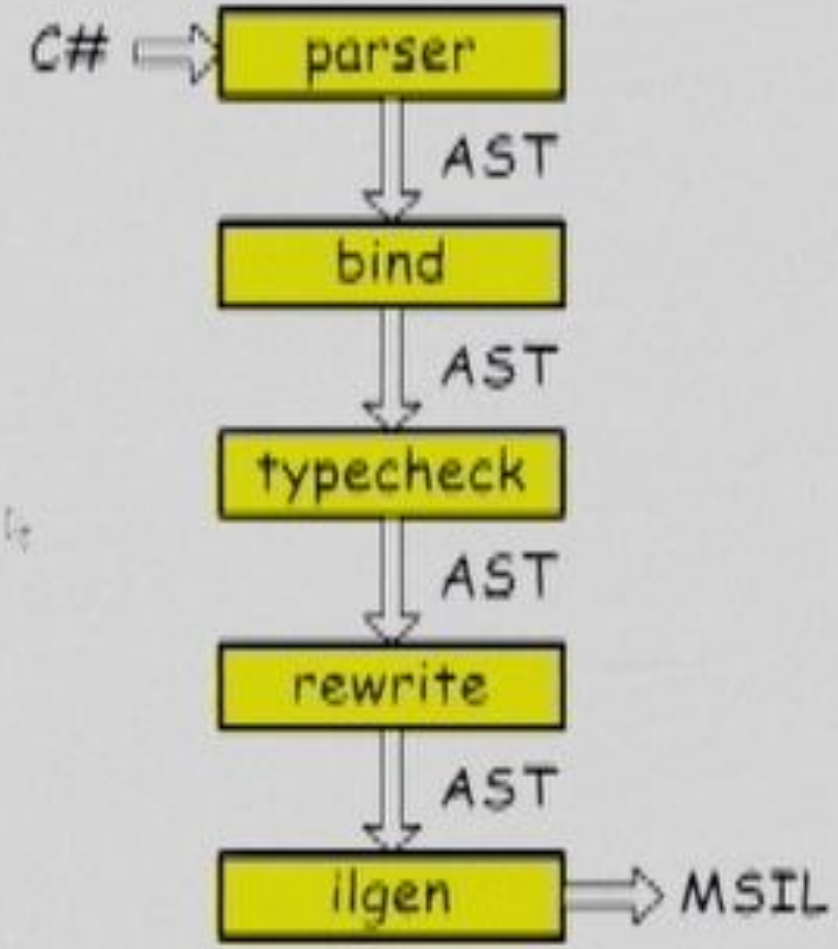
  - Accepts and returns an AST
  - Emit output, if any, to w
- Visitors return the AST, which is passed to others
- Command-line options give visitors, order of invocation

  lcsc -visitor:*classname*,*file* ... *C# source files*
  - Omit *file* if *file* == *classname*
- Visitor class is responsible for traversing the AST
  - mkvisitor generates 800+ lines of skeleton code

    mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
  - [Demo]

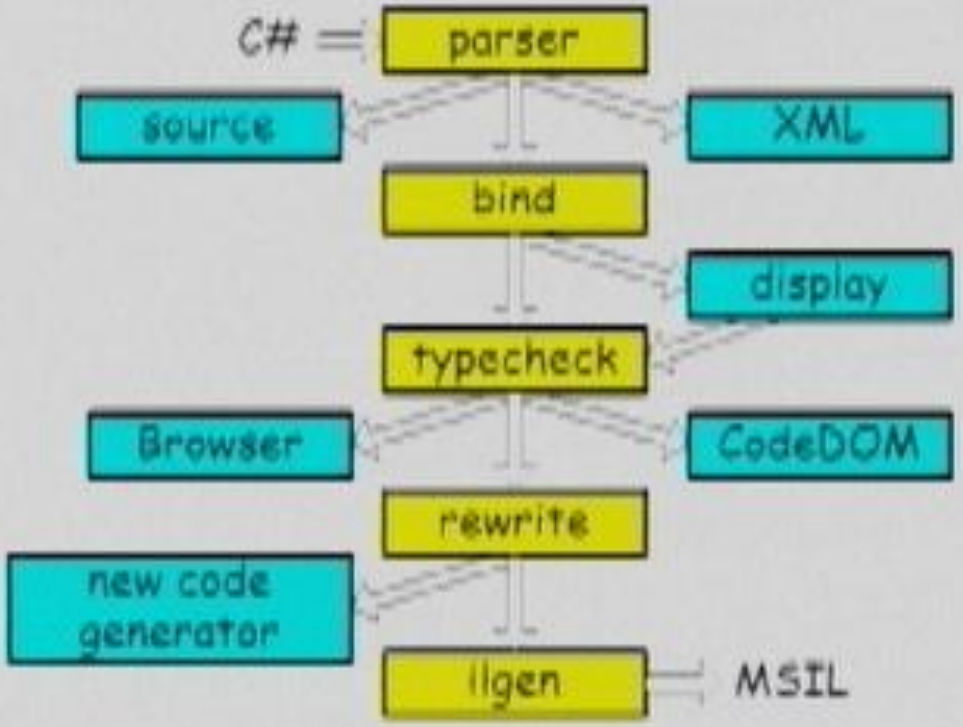
Traditional Compiler Passes

- bind:
  bind uses to definitions,
  annotate with Symbols
- typecheck:
  type check methods,
  annotate with Types,
  Methods
- rewrite:
  prepare for code
  generation
- ilgen:
  emit MSIL
- This is default visitor list



"Exotic" Visitors

- Insert visitors *anywhere*
- XML: emit AST as XML (uses reflection)
- sortmembers: alphabetize declarations
- codedom: emit code to build CodeDOM for C# input
- source: emit C# (also documents ASTs)
- display: display data structures in Internet Explorer
- Browser: navigate C# input, its AST, and its MSIL



Visitor Examples

- Emit XML for hello.cs AST [Demo]

  lcsc -visitor:XML hello.cs

- Pretty print hello.cs [Demo]

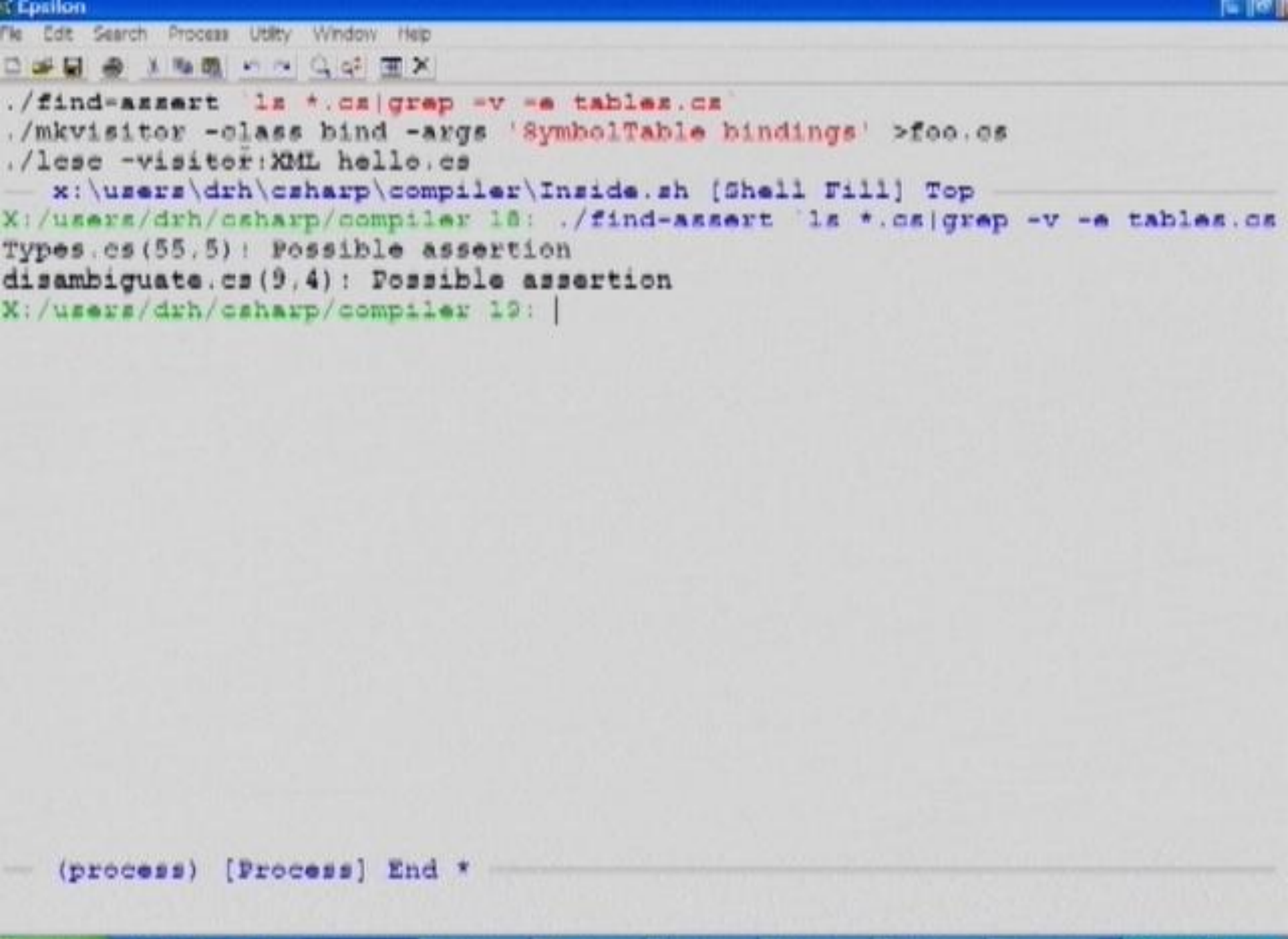
  lcsc -visitor:source hello.cs

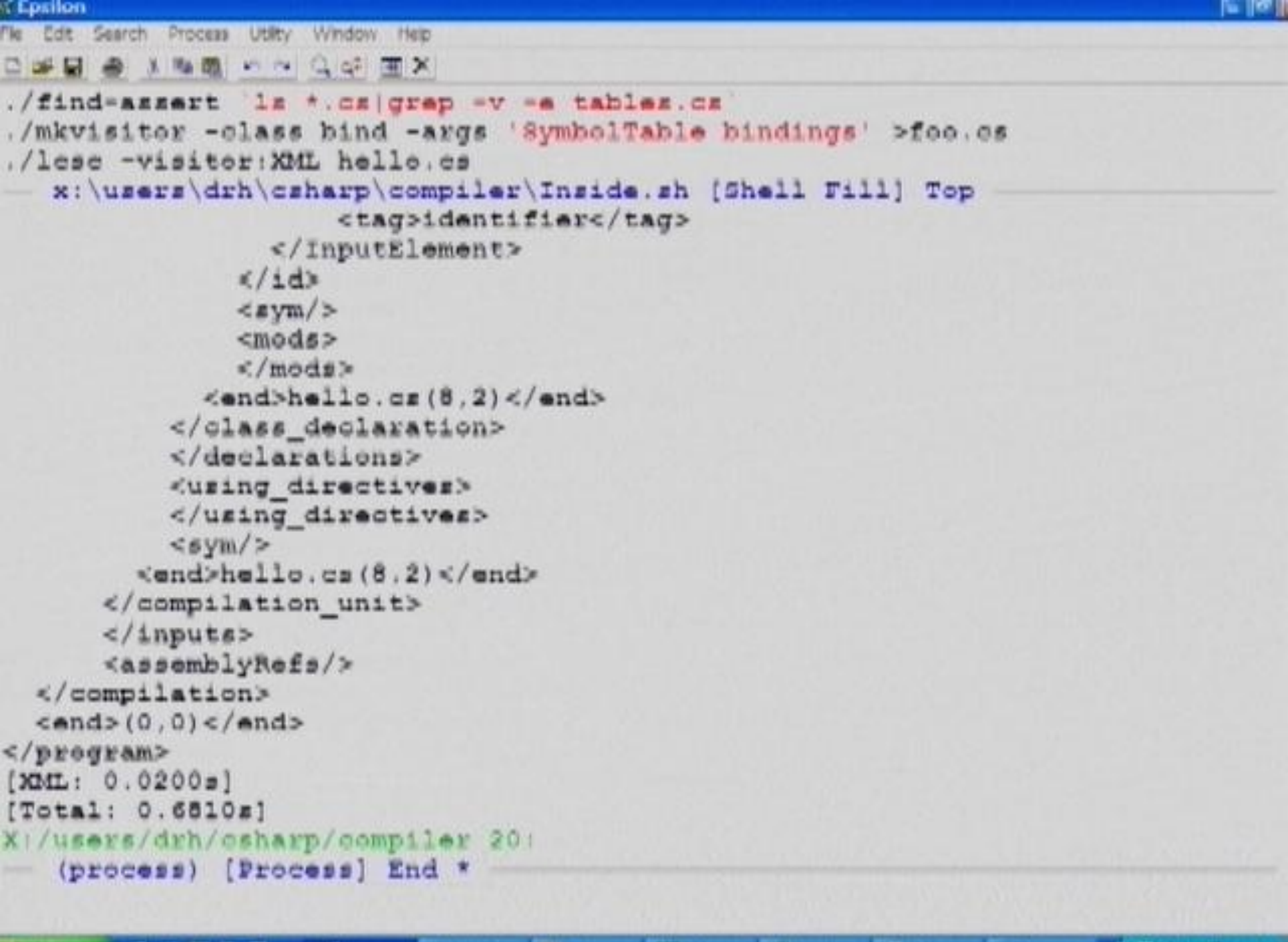
- Alphabetize members in mkvisitor.cs [Demo]

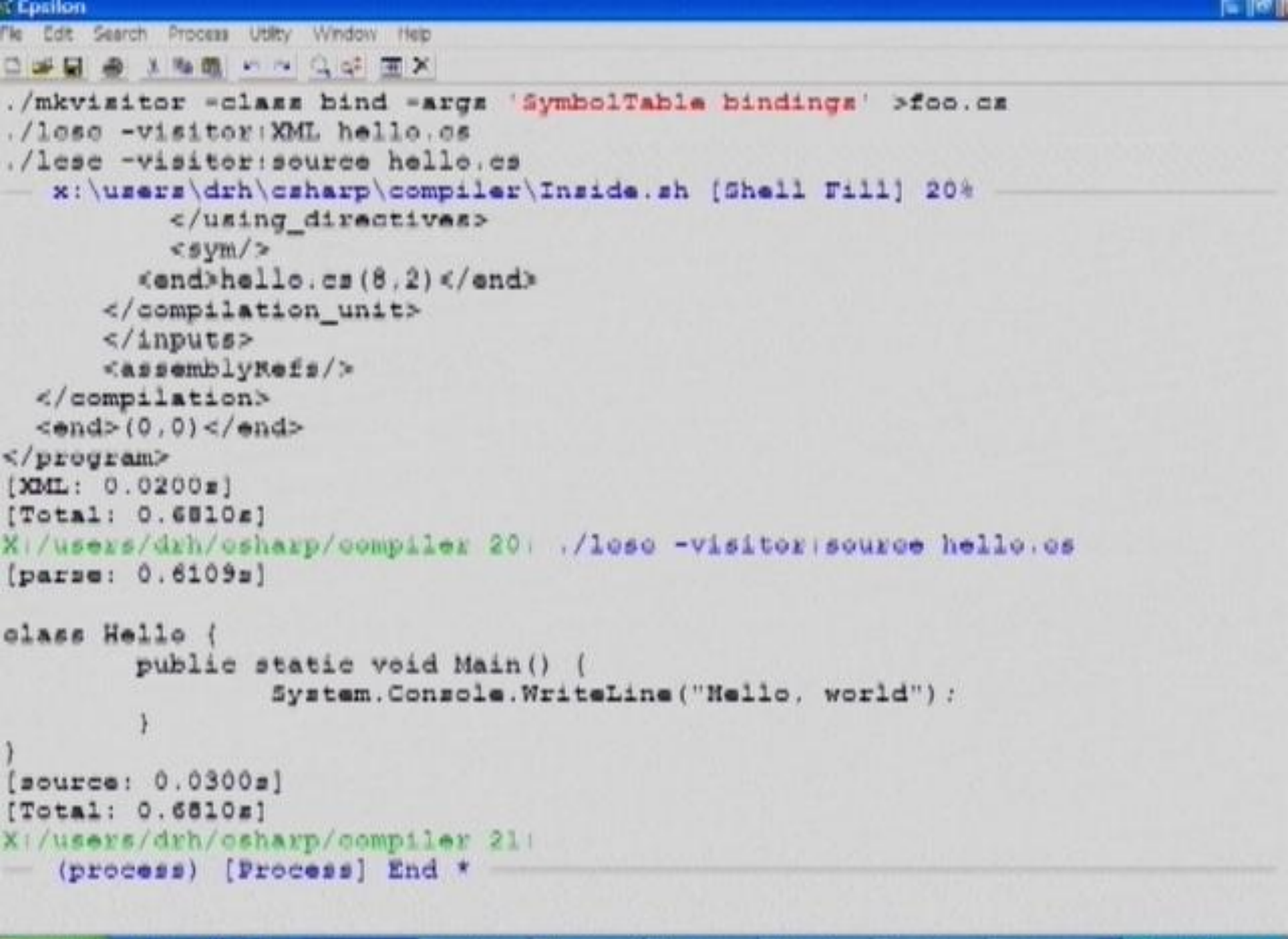
  lcsc -visitor:sortmembers -visitor:source mkvisitor.cs

- Debug bind.cs annotations on hello.cs [Demo]

  lcsc -visitor:bind -visitor:display,debug/base.dll hello.cs


Epsilon
File Edit Search Process Utility Window Help
./find-assert `ls *.cs|grep -v -e tables.cs`
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./lese -visitor:XML hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] Top
X:/users/drh/csharp/compiler 18: ./find-assert `ls *.cs|grep -v -e tables.cs`
Types.cs(55,5): Possible assertion
disambiguate.cs(9,4): Possible assertion
X:/users/drh/csharp/compiler 19:
(process) [Process] End *


Epsilon
File Edit Search Process Utility Window Help
./find-assert 'ls *.cs|grep -v -e tables.cs'
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./lese -visitor:XML hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] Top
<tag>identifier</tag>
</InputElement>
</id>
<sym/>
<mods>
</mods>
<end>hello.cs(8,2)</end>
</class_declaration>
</declarations>
<using_directives>
</using_directives>
<sym/>
<end>hello.cs(8,2)</end>
</compilation_unit>
</inputs>
<assemblyRefs/>
</compilation>
<end>(0,0)</end>
</program>
[XML: 0.0200s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 20:
(process) [Process] End *


Epsilon
File Edit Search Process Utility Window Help
./mkvisitor -class bind -args 'SymbolTable bindings' >foo.cs
./loso -visitor:XML hello.cs
./loso -visitor:source hello.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 20%
</using_directives>
<sym/>
<end>hello.cs(8,2)</end>
</compilation_unit>
</inputs>
<assemblyRefs/>
</compilation>
<end>(0,0)</end>
</program>
[XML: 0.0200s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 20: ./loso -visitor:source hello.cs
[parse: 0.6109s]
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
[source: 0.0300s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 21:
(process) [Process] End *

Visitor Examples

- Emit XML for hello.cs AST [Demo]

  lcsc -visitor:XML hello.cs

- Pretty print hello.cs [Demo]

  lcsc -visitor:source hello.cs

- Alphabetize members in mkvisitor.cs [Demo]

  lcsc -visitor:sortmembers -visitor:source mkvisitor.cs

- Debug bind.cs annotations on hello.cs [Demo]

  lcsc -visitor:bind -visitor:display,debug/base.dll hello.cs

3/7/2003 11:25:52 AM

compilation#7:
    args=stringList{-visitor:bind,-visitor:display,debug/base.dll,hello.cs}
    inputs=compilation_unitList#10
    assemblyRefs=stringList{}
    parent=null
compilation_unitList#10:
    compilation_unit#13
compilation_unit#13:
    attributes=attribute_sectionList#15 (empty)
    declarations=declarationList#17
    using_directives=using_directiveList#19 (empty)
    sym=NameSpace#21
    parent=compilation#7
declarationList#17:
    class_declaration#23
NameSpace#21:
    aliases=stringList{}
    usingdirectives=null

```
class_declaration#23:
    attrs=attribute_sectionList#33 (empty)
    bases=typeList#35 (empty)
    body=declarationList#37
    id=InputElement(coord=hello.cs(3,7),str=Hello,tag=identifier)
    sym=ClassType#43
    mods=InputElementList#45 (empty)
    parent=compilation_unit#13
SymbolTable#27:
    Owner=NameSpace#21
    [System]=NameSpace#49
    [Hello]=ClassType#43
    [Microsoft]=NameSpace#50
declarationList#37:
    method_declaration#51
    constructor_declaration#52
    constructor_declaration#53
ClassType#43:
    baseClass=ClassType#55
    interfaces=InterfaceTypeList#57 (empty)
```

declarationList#37:
    method_declaration#51
    constructor_declaration#52
    constructor_declaration#53
ClassType#43:
    baseClass=ClassType#55
    interfaces=InterfaceTypeList#57 (empty)
    members=SymbolTable#59
    enclosingType=null
    module=null
    attributes=null
    declSpace=SymbolTable#27
    id=InputElement{coord=hello.cs(3,7),str=Hello,tag=identifier}
    modifiers=stringList{internal}
    serial=59
NameSpace#50:
    aliases=stringList{}
    usingdirectives=null
    members=SymbolTable#67
    module=null

```
Method#79:
    interfaceMethod=null
    locals=SymbolTable#127 (empty)
    signature=Signature#129
    module=null
    attributes=null
    declSpace=SymbolTable#59
    id=InputElement{coord=hello.cs(4,21),str=Main,tag=identifier}
    modifiers=stringList{public,static}
    serial=61
InputElementList#81:
    InputElement{coord=hello.cs(4,2),str=public,tag=public}
    InputElement{coord=hello.cs(4,9),str=static,tag=static}
block_statement#86:
    stmts=statementList#136 (empty)
    sym=Block#137
    method=Method#90
    lab=0
    parent=constructor_declaration#52
constructor_declarator#88:
```

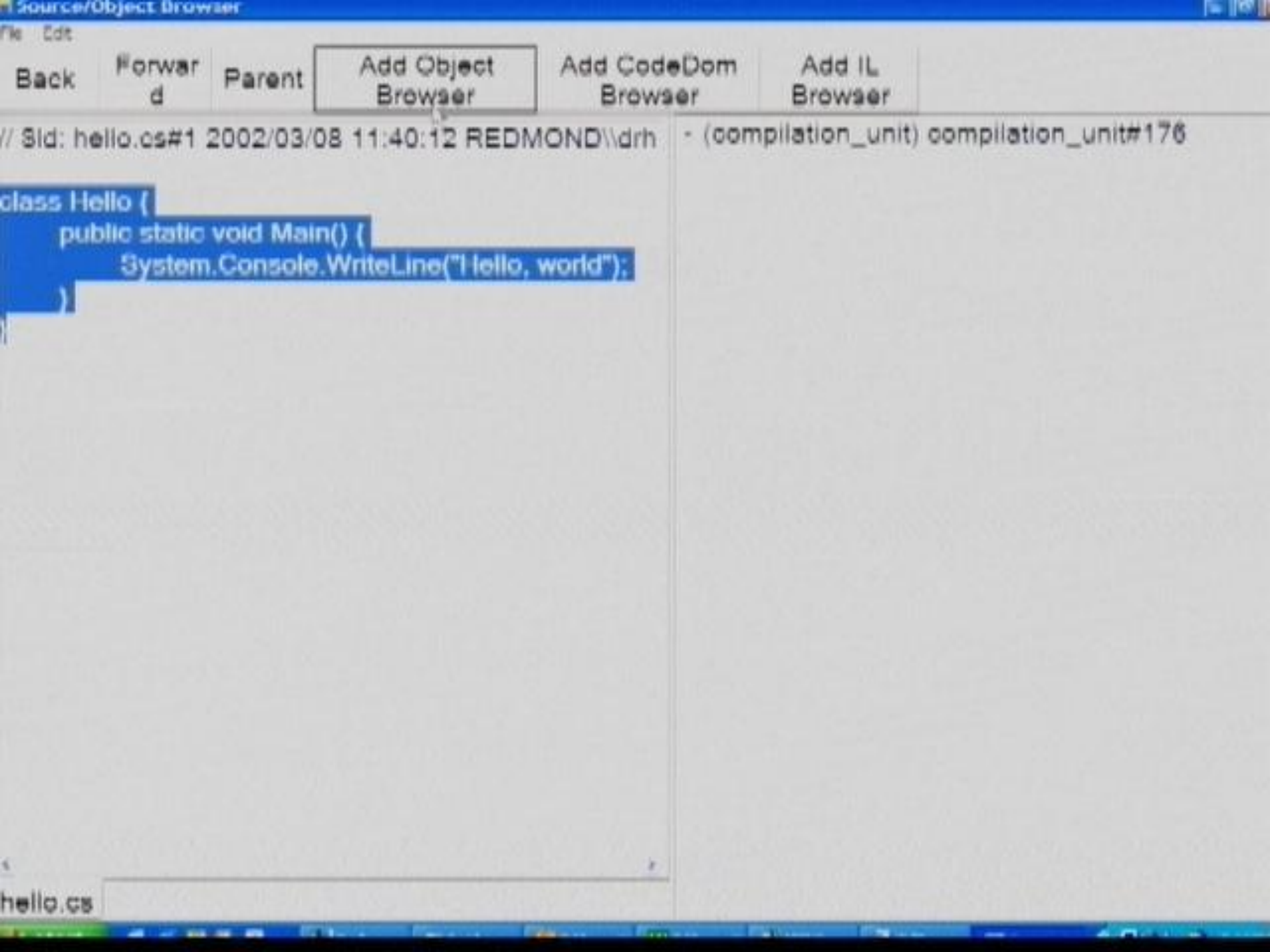
class_declaration#23:
    attrs=attribute_sectionList#33 (empty)
    bases=typeList#35 (empty)
    body=declarationList#37
    id=InputElement(coord=hello.cs(3,7),str=Hello,tag=identifier)
    sym=ClassType#43
    mods=InputElementList#45 (empty)
    parent=compilation_unit#13
SymbolTable#27:
    Owner=NameSpace#21
    [System]=NameSpace#49
    [Hello]=ClassType#43
    [Microsoft]=NameSpace#50
declarationList#37:
    method_declaration#51
    constructor_declaration#52
    constructor_declaration#53
ClassType#43:
    baseClass=ClassType#55
    interfaces=InterfaceTypeList#57 (empty)

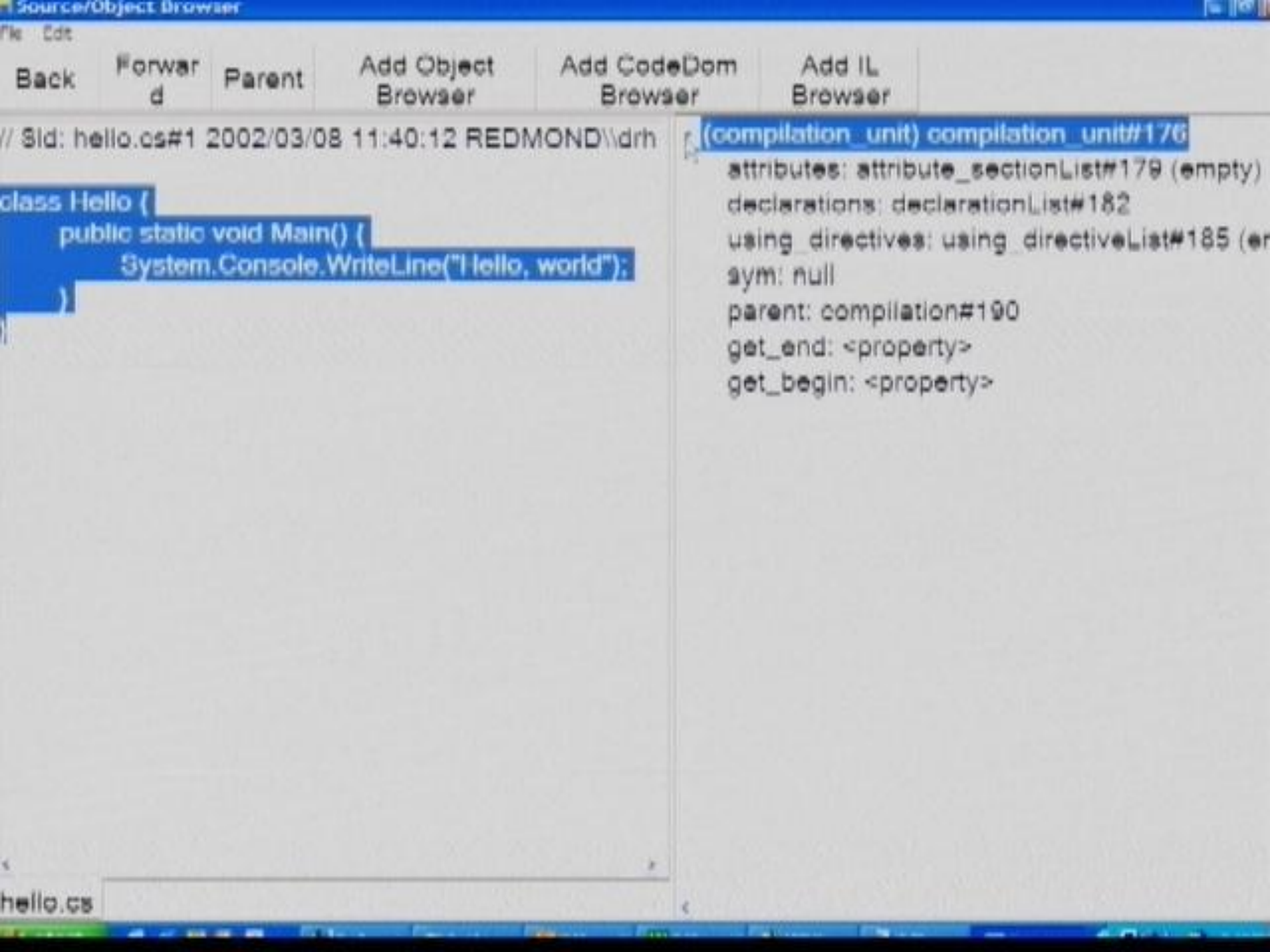
Navigating C# Source and ASTs

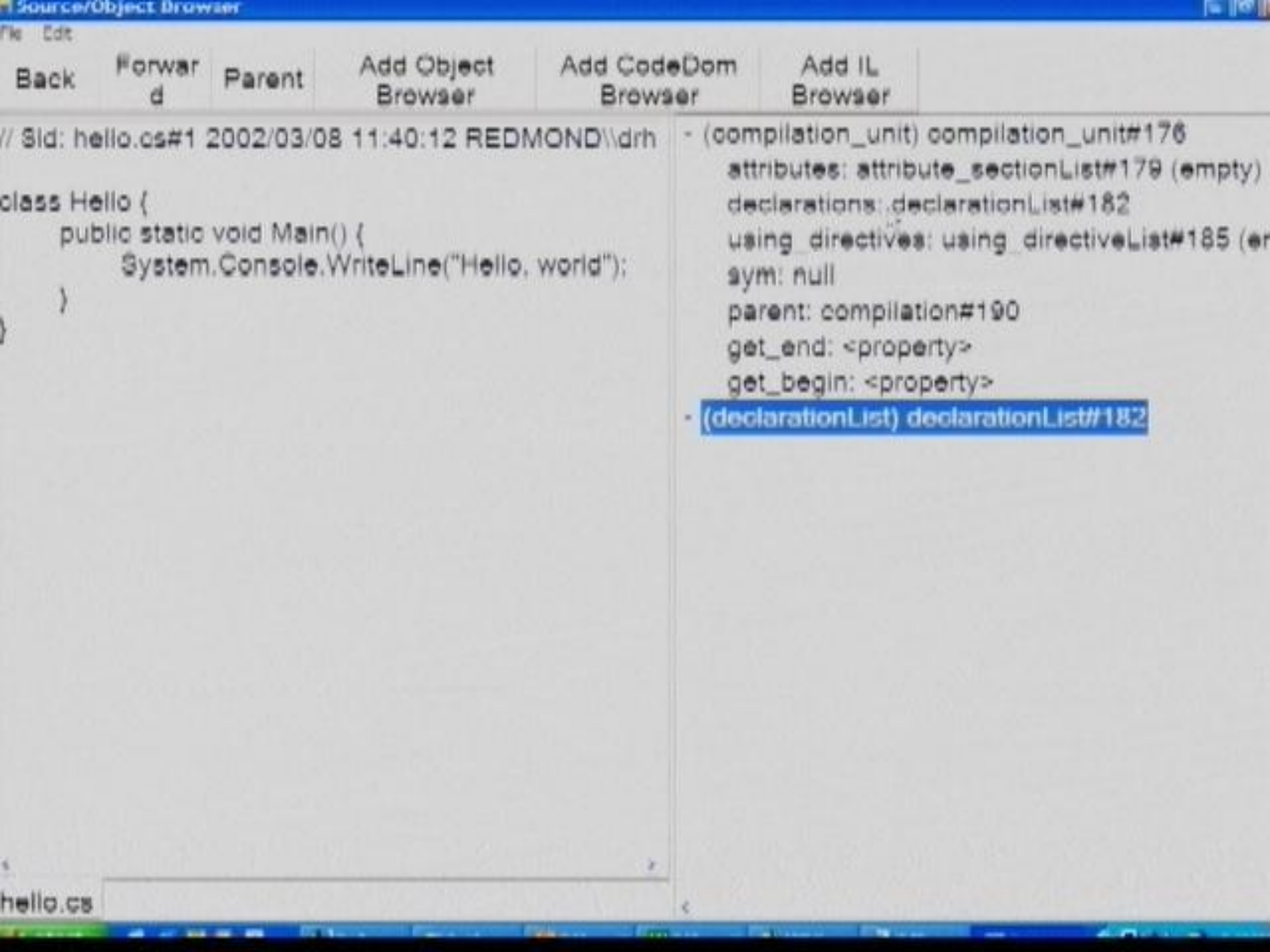
- Display visitor shows too much—6MB is typical
- Hard to correlate ASTs with corresponding source
- Todd's Browser visitor [Demo]
  - Reveals just those data structures you request
  - Correlates AST nodes to source

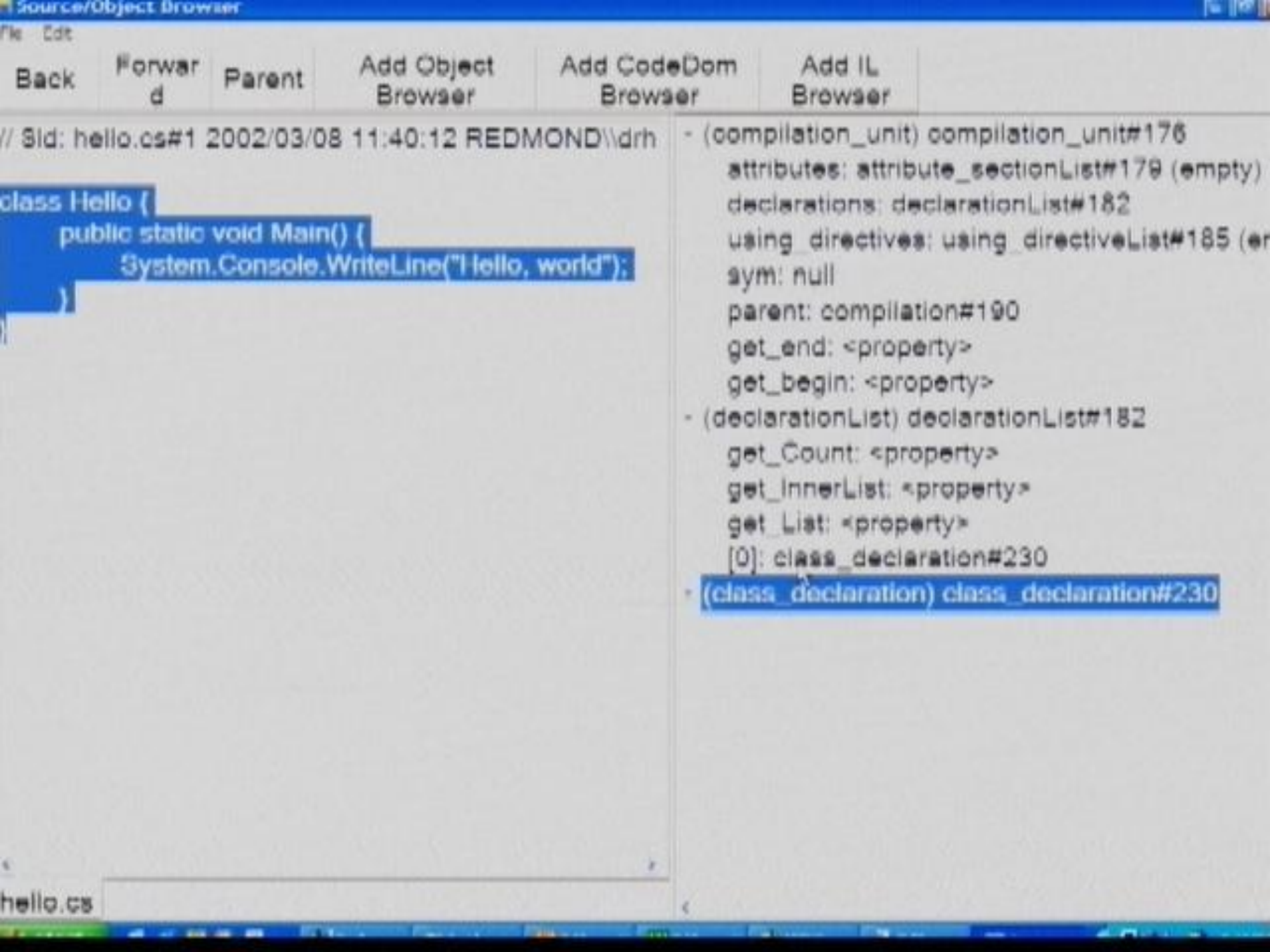
  ```
  lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
  ```
- Ditto for generated MSIL [Demo]

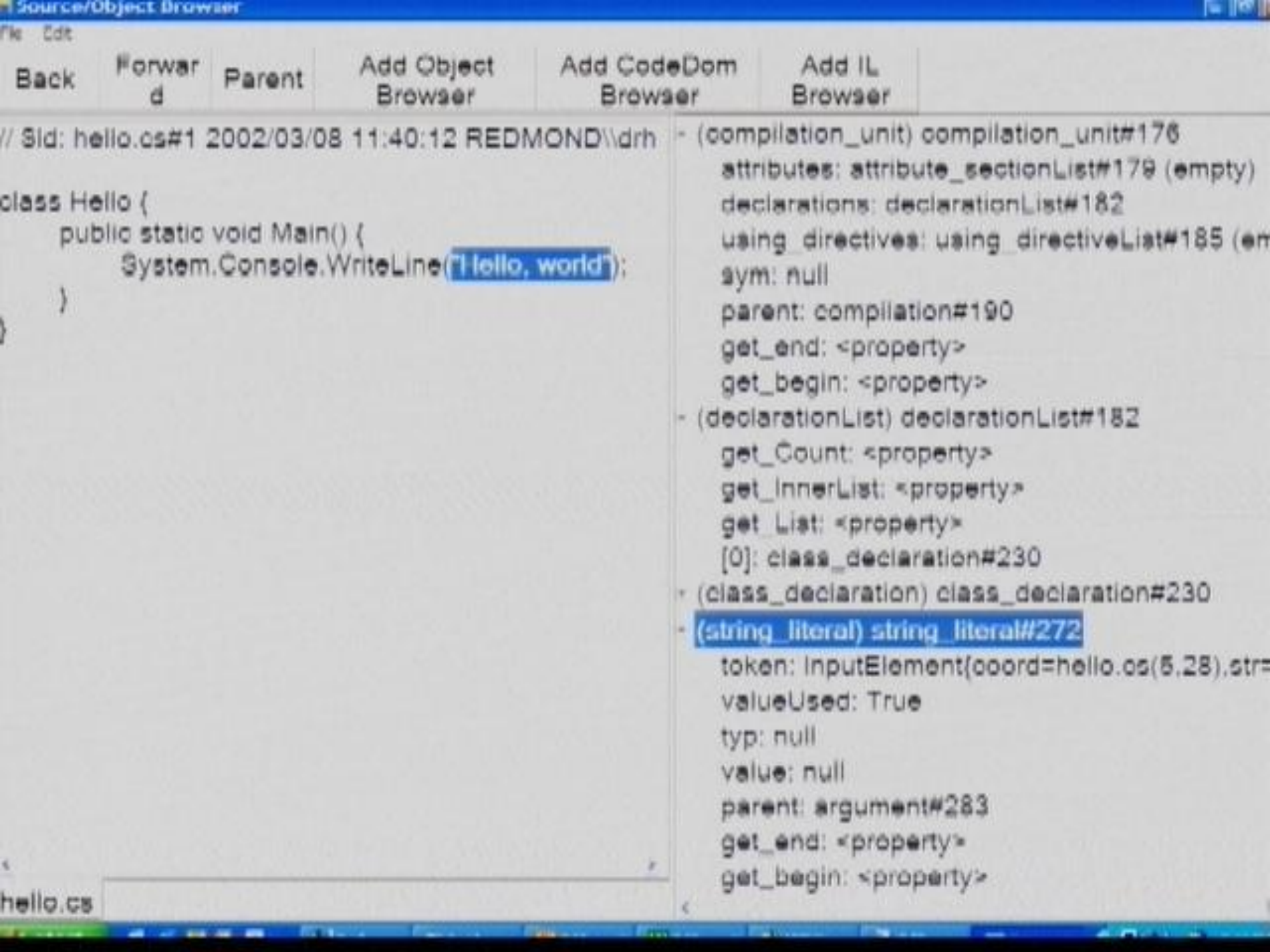
  ```
  lcsc -visitor:bind -visitor:typecheck
      -visitor:Browser.BrowserVisitor,debug/Browser.dll 8q.cs
  ```
- Ditto for CodeDOM

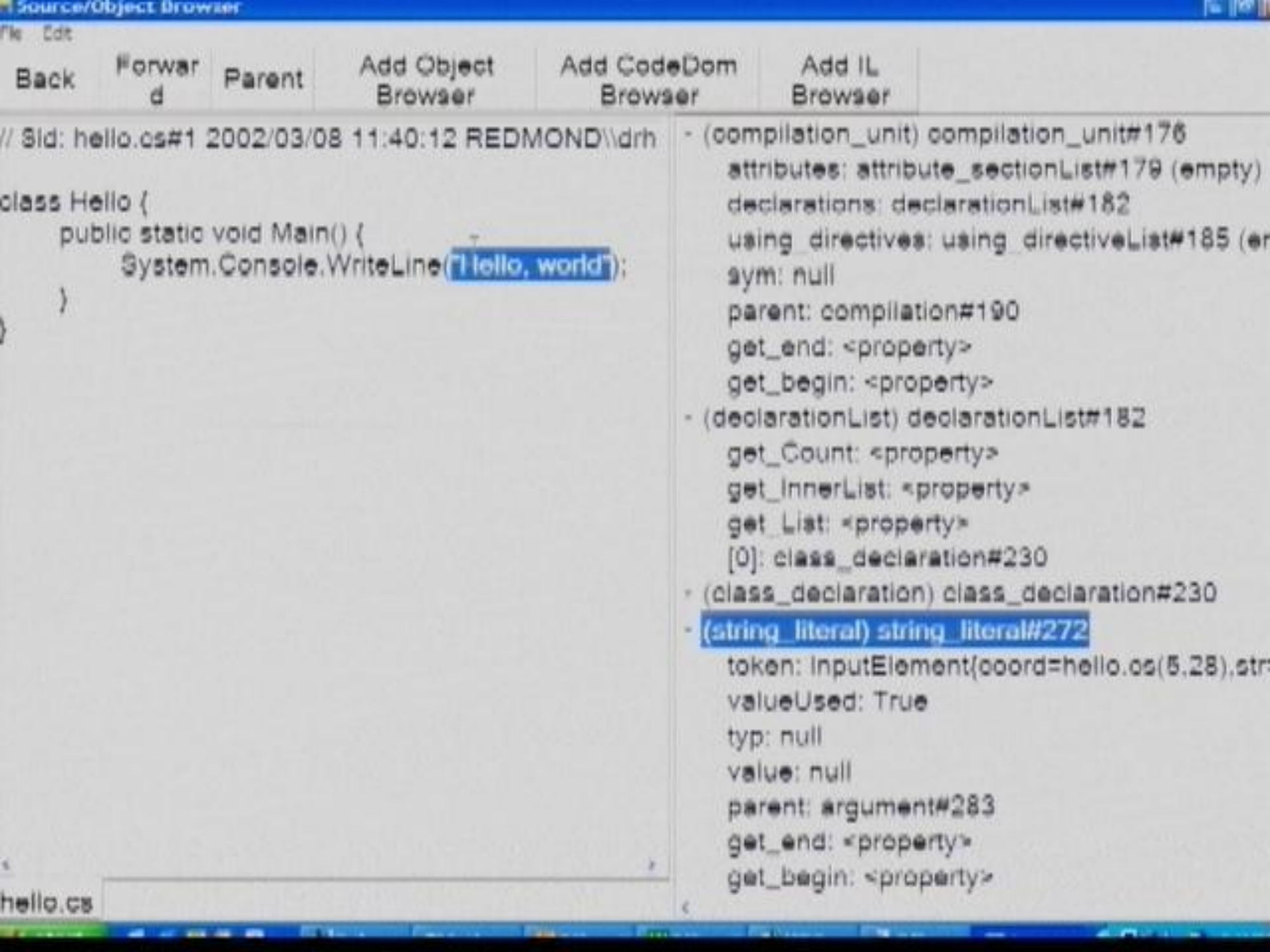

Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
- (compilation_unit) compilation_unit#176
hello.cs

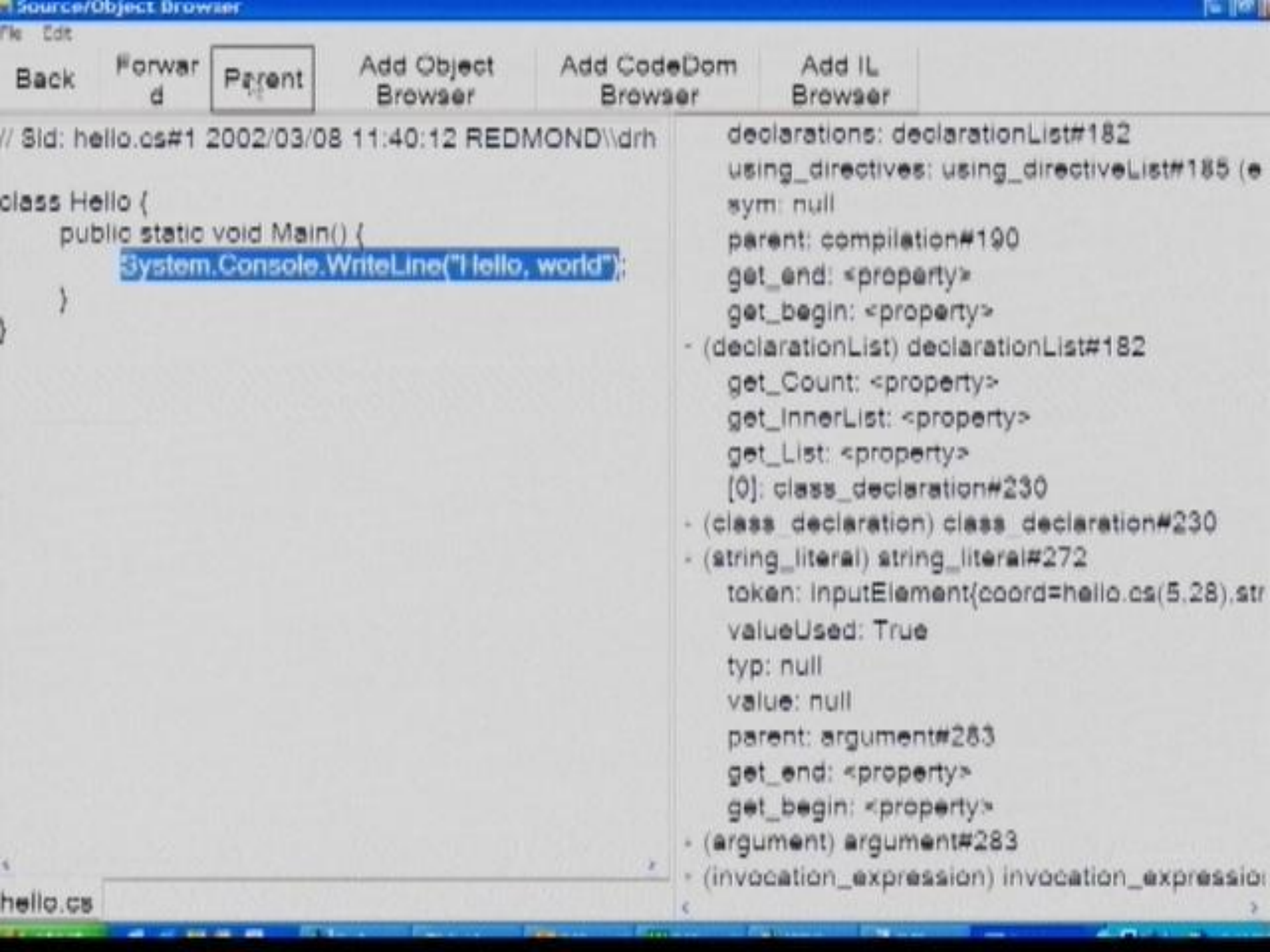

Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
hello.cs
(compilation_unit) compilation_unit#176
attributes: attribute_sectionList#179 (empty)
declarations: declarationList#182
using_directives: using_directiveList#185 (e
sym: null
parent: compilation#190
get_end: <property>
get_begin: <property>


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
hello.cs
- (compilation_unit) compilation_unit#176
attributes: attribute_sectionList#179 (empty)
declarations: declarationList#182
using_directives: using_directiveList#185 (e
sym: null
parent: compilation#190
get_end: <property>
get_begin: <property>
- (declarationList) declarationList#182


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
hello.cs
- (compilation_unit) compilation_unit#176
attributes: attribute_sectionList#178 (empty)
declarations: declarationList#182
using_directives: using_directiveList#185 (em
sym: null
parent: compilation#190
get_end: <property>
get_begin: <property>
- (declarationList) declarationList#182
get_Count: <property>
get_InnerList: <property>
get_List: <property>
[0]: class_declaration#230
- (class_declaration) class_declaration#230


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
hello.cs
- (compilation_unit) compilation_unit#176
attributes: attribute_sectionList#179 (empty)
declarations: declarationList#182
using_directives: using_directiveList#185 (em
sym: null
parent: compilation#190
get_end: <property>
get_begin: <property>
- (declarationList) declarationList#182
get_Count: <property>
get_InnerList: <property>
get_List: <property>
[0]: class_declaration#230
- (class_declaration) class_declaration#230
- (string_literal) string_literal#272
token: InputElement(coord=hello.cs(5,28),str
valueUsed: True
typ: null
value: null
parent: argument#283
get_end: <property>
get_begin: <property>


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
hello.cs
- (compilation_unit) compilation_unit#176
attributes: attribute_sectionList#179 (empty)
declarations: declarationList#182
using_directives: using_directiveList#185 (em
sym: null
parent: compilation#190
get_end: <property>
get_begin: <property>
- (declarationList) declarationList#182
get_Count: <property>
get_InnerList: <property>
get_List: <property>
[0]: class_declaration#230
- (class_declaration) class_declaration#230
- (string_literal) string_literal#272
token: InputElement(coord=hello.cs(5,28),str
valueUsed: True
typ: null
value: null
parent: argument#283
get_end: <property>
get_begin: <property>

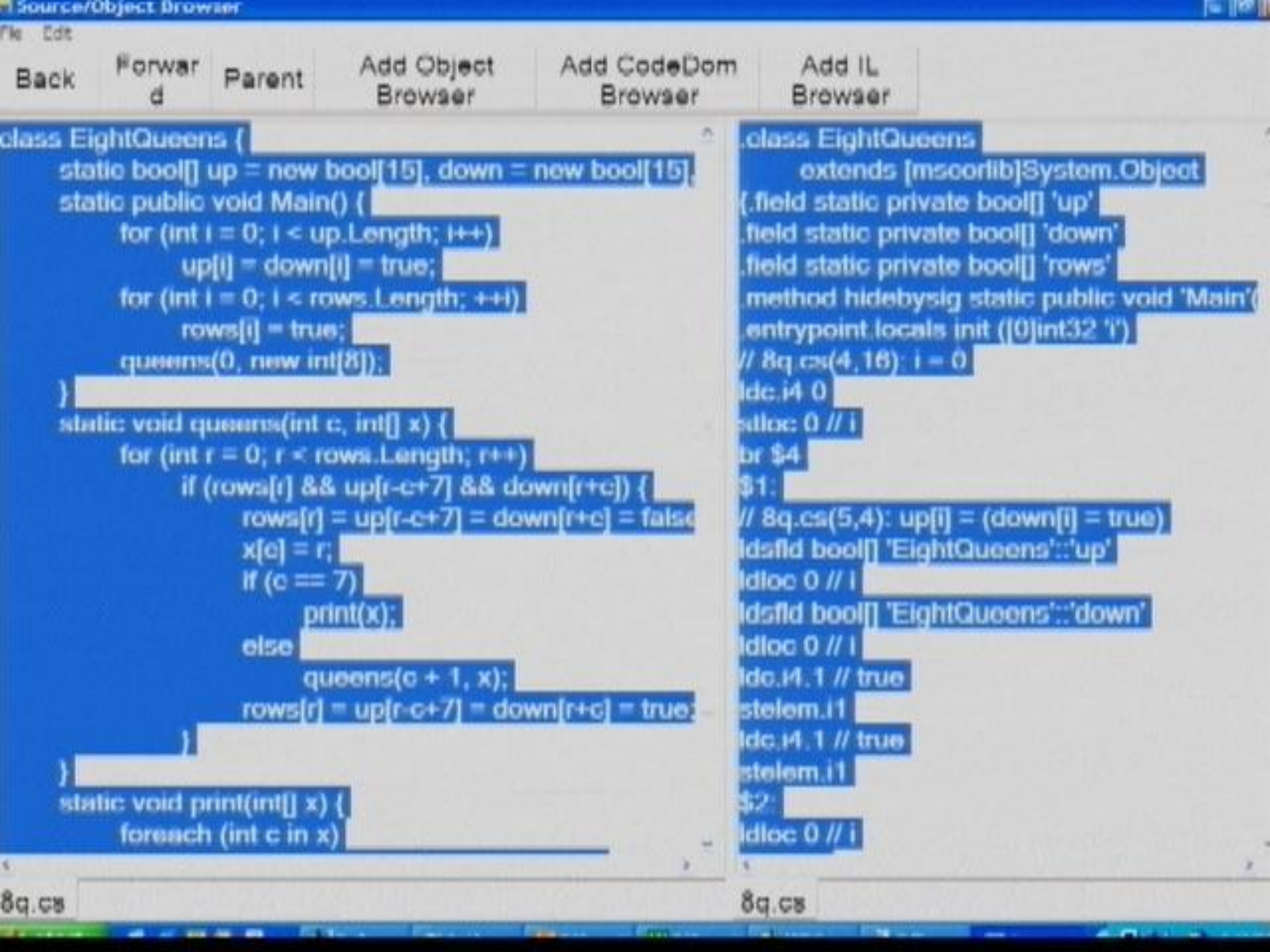

Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
// $Id: hello.cs#1 2002/03/08 11:40:12 REDMOND\\drh
class Hello {
public static void Main() {
System.Console.WriteLine("I lello, world");
}
}
declarations: declarationList#182
using_directives: using_directiveList#185 (e
sym: null
parent: compilation#190
get_end: <property>
get_begin: <property>
- (declarationList) declarationList#182
get_Count: <property>
get_InnerList: <property>
get_List: <property>
[0]: class_declaration#230
- (class_declaration) class_declaration#230
- (string_literal) string_literal#272
token: InputElement{coord=hello.cs(5,28),str
valueUsed: True
typ: null
value: null
parent: argument#283
get_end: <property>
get_begin: <property>
- (argument) argument#283
- (invocation_expression) invocation_expressio
hello.cs

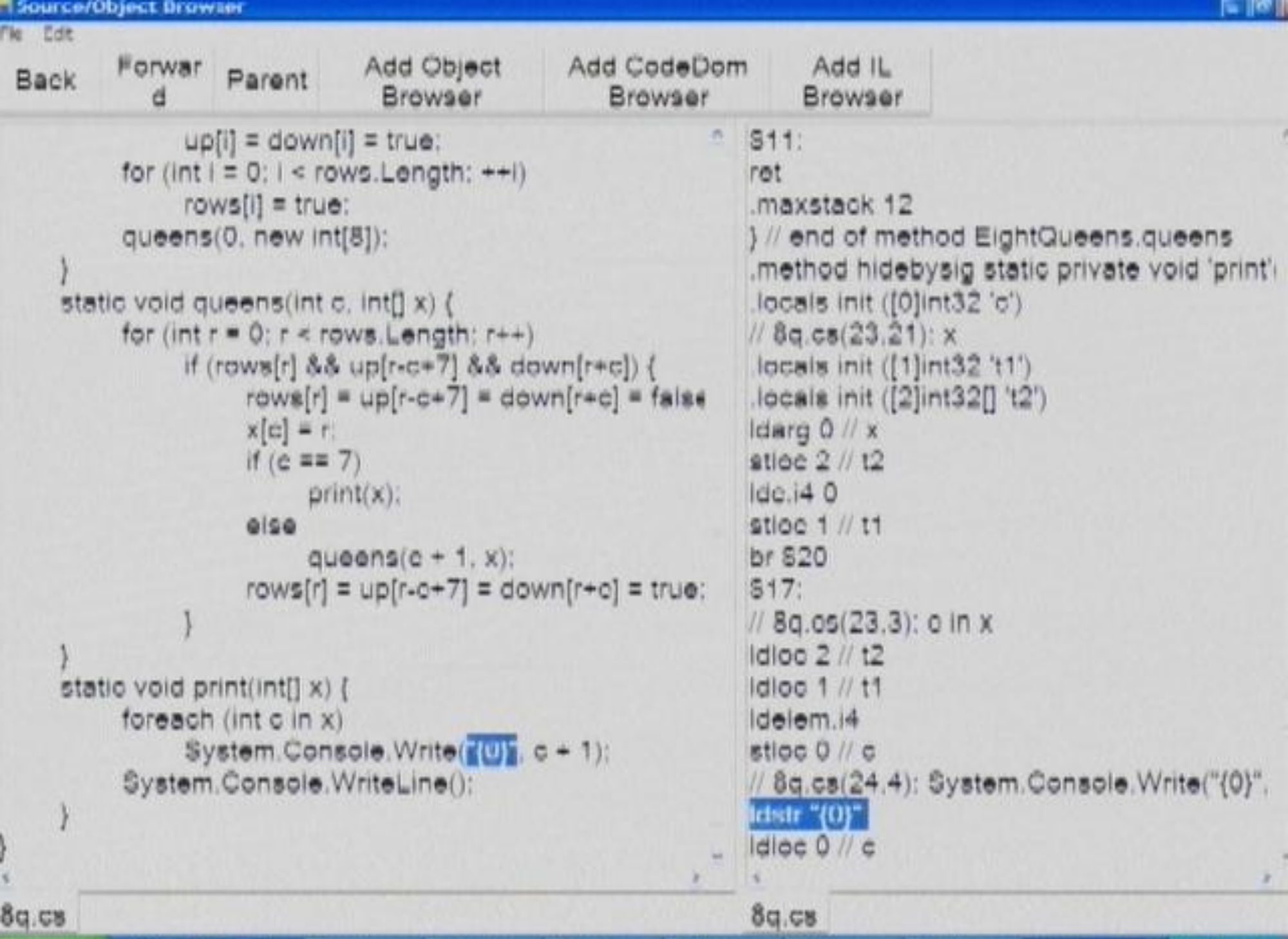
Navigating C# Source and ASTs

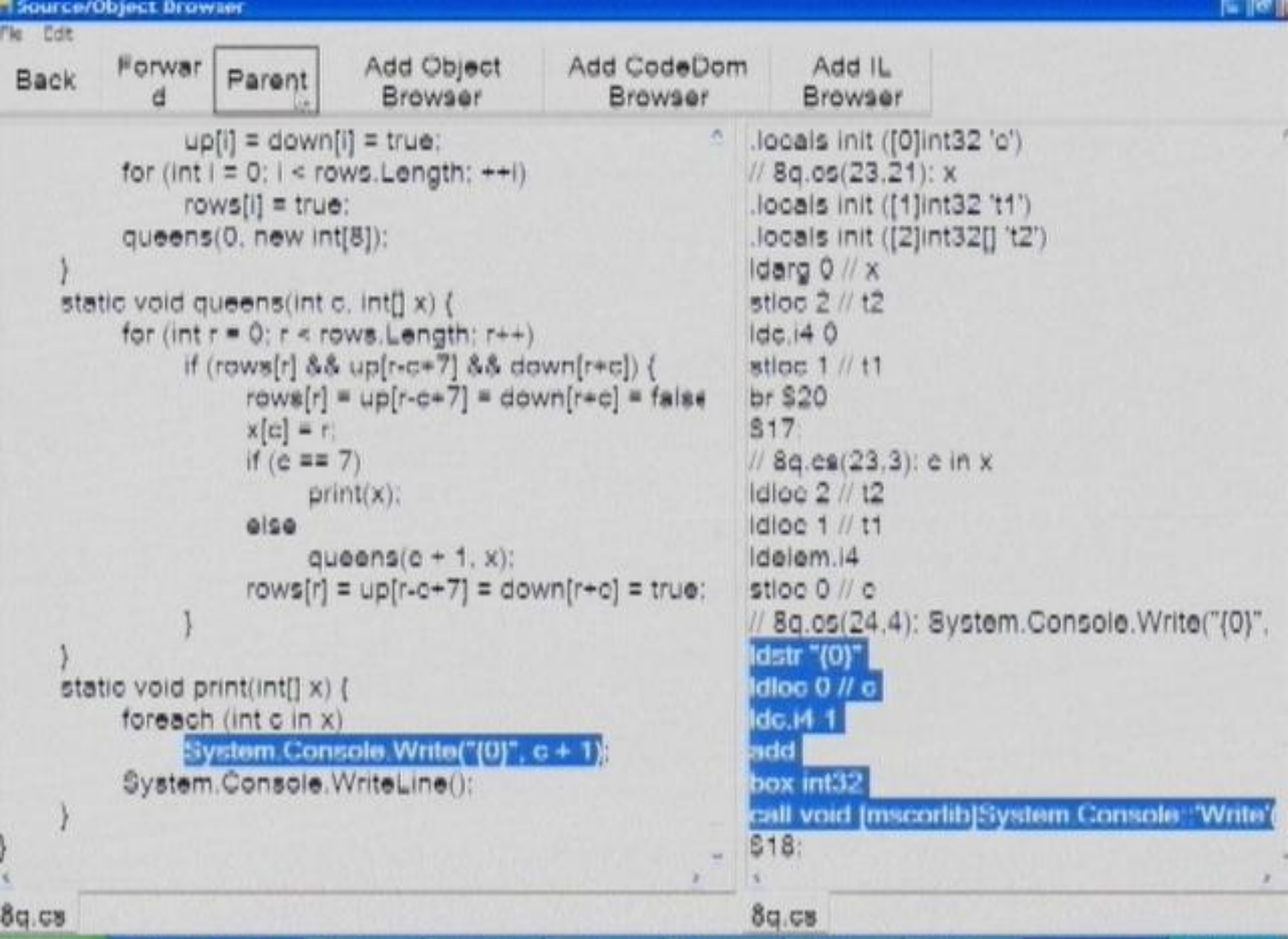
- Display visitor shows too much—6MB is typical
- Hard to correlate ASTs with corresponding source
- Todd's Browser visitor [Demo]
  - Reveals just those data structures you request
  - Correlates AST nodes to source

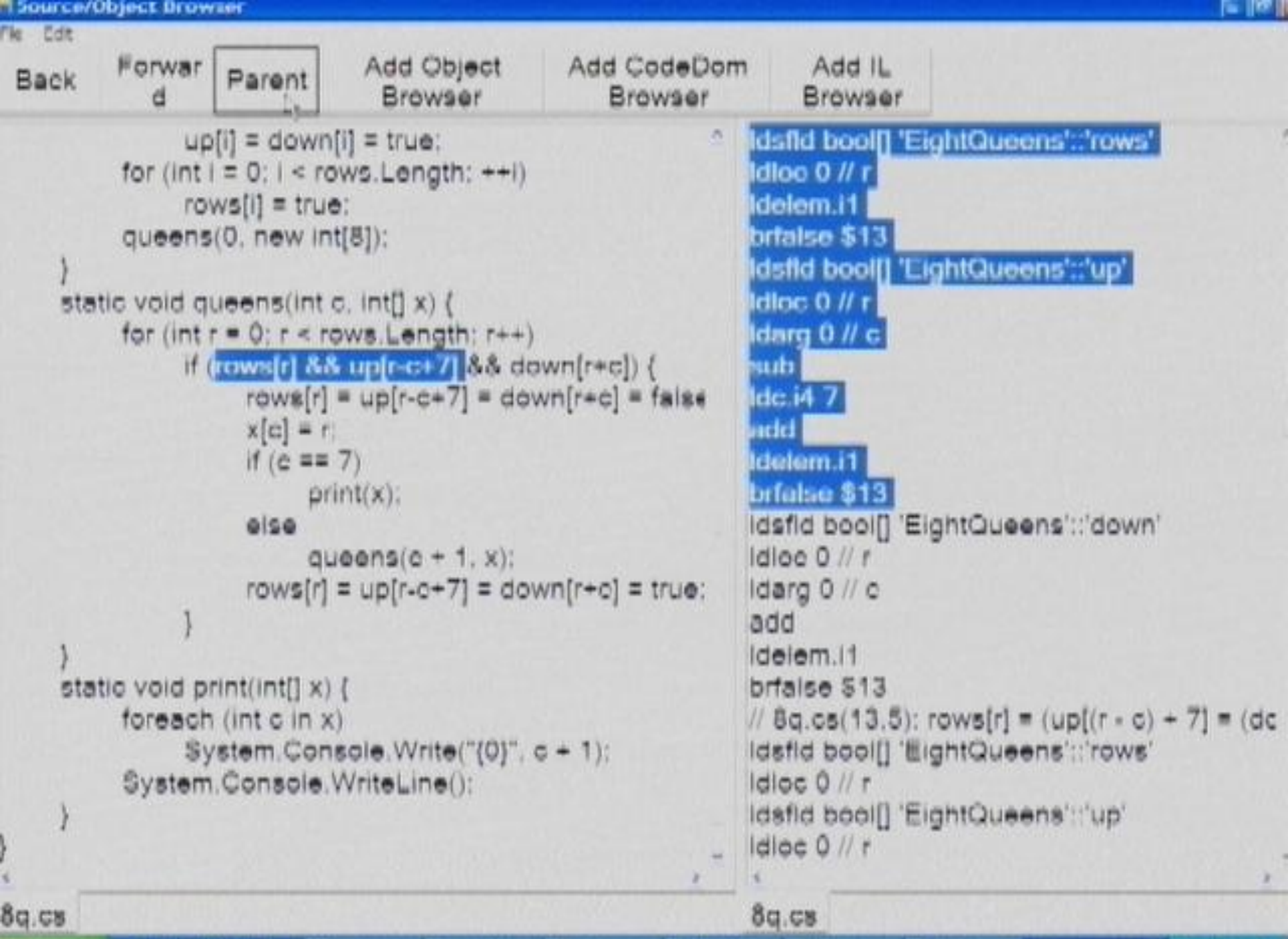
  ```
  lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
  ```
- Ditto for generated MSIL [Demo]

  ```
  lcsc -visitor:bind -visitor:typecheck
      -visitor:Browser.BrowserVisitor,debug/Browser.dll Bq.cs
  ```
- Ditto for CodeDOM


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
class EightQueens {
static bool[] up = new bool[15], down = new bool[15],
static public void Main() {
for (int i = 0; i < up.Length; i++)
up[i] = down[i] = true;
for (int i = 0; i < rows.Length; ++i)
rows[i] = true;
queens(0, new int[8]);
}
static void queens(int c, int[] x) {
for (int r = 0; r < rows.Length; r++)
if (rows[r] && up[r-c+7] && down[r+c]) {
rows[r] = up[r-c+7] = down[r+c] = fals
x[c] = r;
if (c == 7)
print(x);
else
queens(c + 1, x);
rows[r] = up[r-c+7] = down[r+c] = true;
}
}
static void print(int[] x) {
foreach (int c in x)
8q.cs
.class EightQueens
extends [mscorlib]System.Object
{.field static private bool[] 'up'
.field static private bool[] 'down'
.field static private bool[] 'rows'
.method hidebysig static public void 'Main'(
.entrypoint.locals init ([0]int32 'i')
// 8q.cs(4,16): i = 0
ldc.i4 0
stloc 0 // i
br $4
$1:
// 8q.cs(5,4): up[i] = (down[i] = true)
ldsfld bool[] 'EightQueens'::'up'
ldloc 0 // i
ldsfld bool[] 'EightQueens'::'down'
ldloc 0 // i
ldc.i4.1 // true
stelem.i1
ldc.i4.1 // true
stelem.i1
$2:
ldloc 0 // i
8q.cs


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
up[i] = down[i] = true;
for (int i = 0; i < rows.Length; ++i)
rows[i] = true;
queens(0, new int[8]);
}
static void queens(int c, int[] x) {
for (int r = 0; r < rows.Length; r++)
if (rows[r] && up[r-c+7] && down[r+c]) {
rows[r] = up[r-c+7] = down[r+c] = false
x[c] = r;
if (c == 7)
print(x);
else
queens(c + 1, x);
rows[r] = up[r-c+7] = down[r+c] = true;
}
}
static void print(int[] x) {
foreach (int c in x)
System.Console.Write("{0}", c + 1);
System.Console.WriteLine();
}
8q.cs
IL_0011:
ret
.maxstack 12
} // end of method EightQueens::queens
.method hidebysig static private void 'print'(
.locals init ([0]int32 'c')
// 8q.cs(23,21): x
.locals init ([1]int32 't1')
.locals init ([2]int32[] 't2')
ldarg 0 // x
stloc 2 // t2
ldc.i4 0
stloc 1 // t1
br IL_0020
IL_0017:
// 8q.cs(23,3): c in x
ldloc 2 // t2
ldloc 1 // t1
ldelem.i4
stloc 0 // c
// 8q.cs(24,4): System.Console.Write("{0}",
ldstr "{0}"
ldloc 0 // c
8q.cs


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
up[i] = down[i] = true;
for (int i = 0; i < rows.Length; ++i)
rows[i] = true;
queens(0, new int[8]);
}
static void queens(int c, int[] x) {
for (int r = 0; r < rows.Length; r++)
if (rows[r] && up[r-c+7] && down[r+c]) {
rows[r] = up[r-c+7] = down[r+c] = false
x[c] = r;
if (c == 7)
print(x);
else
queens(c + 1, x);
rows[r] = up[r-c+7] = down[r+c] = true;
}
}
static void print(int[] x) {
foreach (int c in x)
System.Console.Write("{0}", c + 1);
System.Console.WriteLine();
}
}
8q.cs
.locals init ([0]int32 'c')
// 8q.cs(23,21): x
.locals init ([1]int32 't1')
.locals init ([2]int32[] 't2')
ldarg 0 // x
stloc 2 // t2
ldc.i4 0
stloc 1 // t1
br $20
$17:
// 8q.cs(23,3): c in x
ldloc 2 // t2
ldloc 1 // t1
ldelem.i4
stloc 0 // c
// 8q.cs(24,4): System.Console.Write("{0}",
ldstr "{0}"
ldloc 0 // c
ldc.i4 1
add
box int32
call void [mscorlib]System.Console::'Write'(
$18:
8q.cs


Source/Object Browser
File Edit
Back
Forward
Parent
Add Object Browser
Add CodeDom Browser
Add IL Browser
up[i] = down[i] = true;
for (int i = 0; i < rows.Length; ++i)
rows[i] = true;
queens(0, new int[8]);
}
static void queens(int c, int[] x) {
for (int r = 0; r < rows.Length; r++)
if (rows[r] && up[r-c+7] && down[r+c]) {
rows[r] = up[r-c+7] = down[r+c] = false
x[c] = r;
if (c == 7)
print(x);
else
queens(c + 1, x);
rows[r] = up[r-c+7] = down[r+c] = true;
}
}
static void print(int[] x) {
foreach (int c in x)
System.Console.Write("{0}", c + 1);
System.Console.WriteLine();
}
8q.cs
ldsfld bool[] 'EightQueens'::'rows'
ldloc 0 // r
ldelem.i1
brfalse $13
ldsfld bool[] 'EightQueens'::'up'
ldloc 0 // r
ldarg 0 // c
sub
ldc.i4 7
add
ldelem.i1
brfalse $13
ldsfld bool[] 'EightQueens'::'down'
ldloc 0 // r
ldarg 0 // c
add
ldelem.i1
brfalse $13
// 8q.cs(13,5): rows[r] = (up[(r - c) + 7] = (dc
ldsfld bool[] 'EightQueens'::'rows'
ldloc 0 // r
ldsfld bool[] 'EightQueens'::'up'
ldloc 0 // r
8q.cs

Navigating C# Source and ASTs

- Display visitor shows too much—6MB is typical
- Hard to correlate ASTs with corresponding source
- Todd's Browser visitor [Demo]
  - Reveals just those data structures you request
  - Correlates AST nodes to source

  ```
  lcsc -visitor:Browser,BrowserVisitor,debug/Browser.dll hello.cs
  ```
- Ditto for generated MSIL [Demo]

  ```
  lcsc -visitor:bind -visitor:typecheck
      -visitor:Browser,BrowserVisitor,debug/Browser.dll 8q.cs
  ```
- Ditto for CodeDOM

Extending C#: Adding typeswitch

- typeswitch statement, ala MODULA-3's TYPECASE

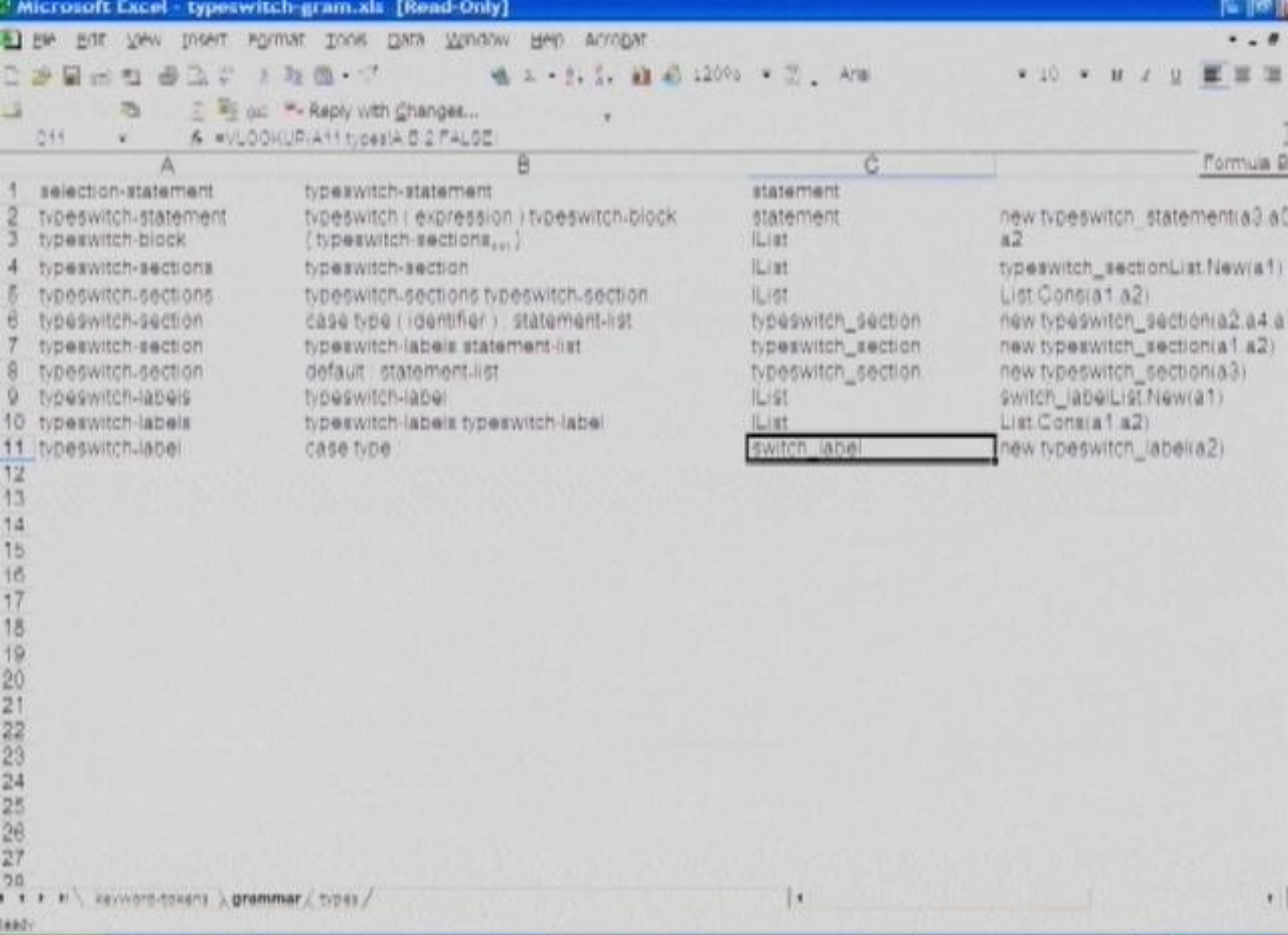
```
typeswitch (o) {
   case Int32  (x): Console.WriteLine(x); break;
   case Symbol (s): Symbols.Add(s); break;
   case Segment: popSegment(); break;
   default: throw new ArgumentException();
}
```

- Abbreviates if-else chains: introduces locals, performs downcasts

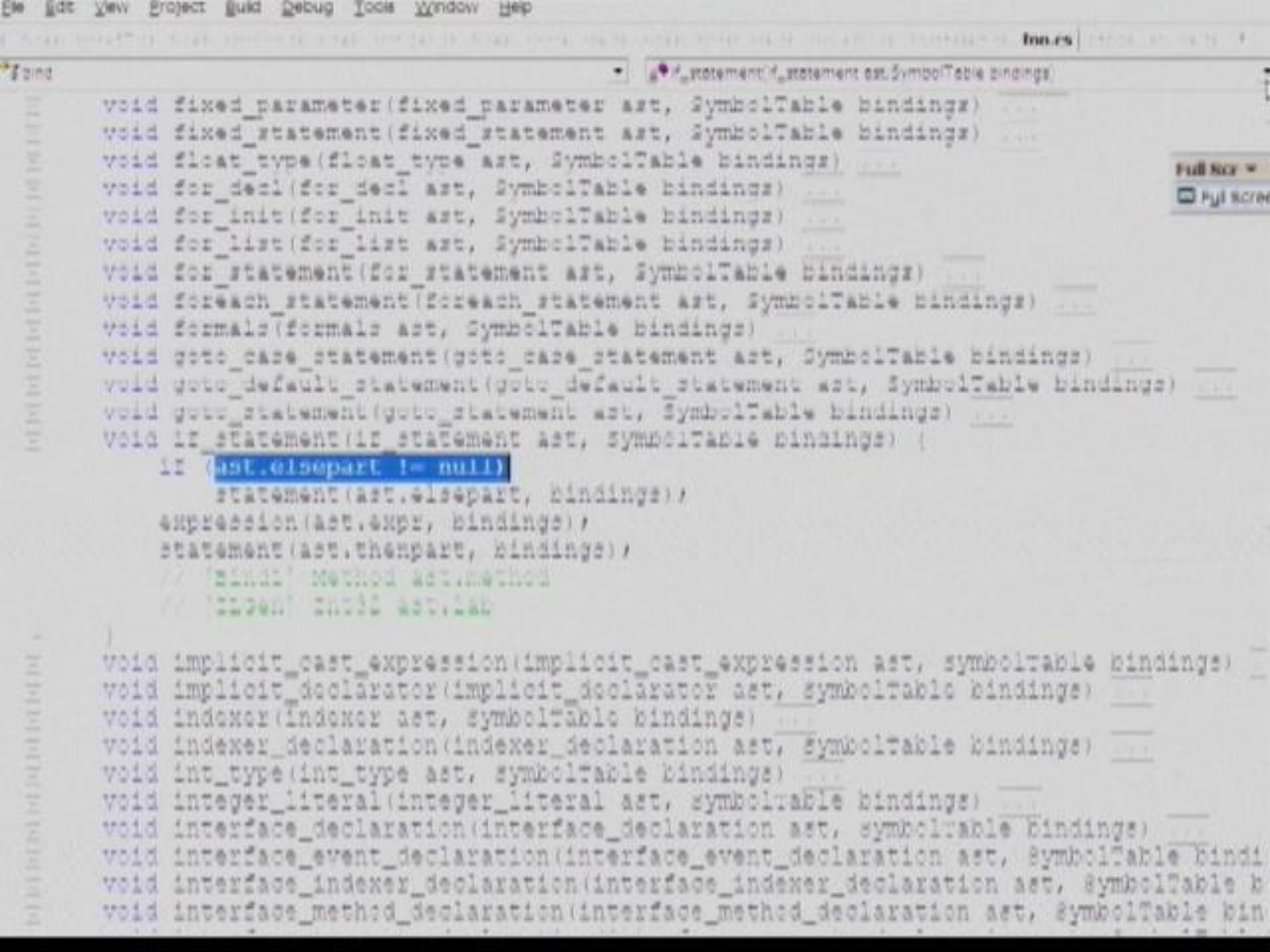
```
if (o is Int32) {
   int x = (int)o; Console.WriteLine(x); }
else if (o is Symbol) {
   Symbol s = (Symbol)o; Symbols.Add(s); }
else if (o is Segment) { popSegment(); }
else { throw new InvalidArgument(); }
```

The Easy Way: Source-to-Source Transformation

- Prepare grammar spreadsheet (1+11+7=19 lines) [Demo]
- Add typeswitch AST nodes (63 lines) [Demo]
- *Extend* source.cs (71 lines) [Demo]
- Alternative
  - Build appropriate if-else ASTs directly, or rewrite AST
  - Use source.cs unmodified



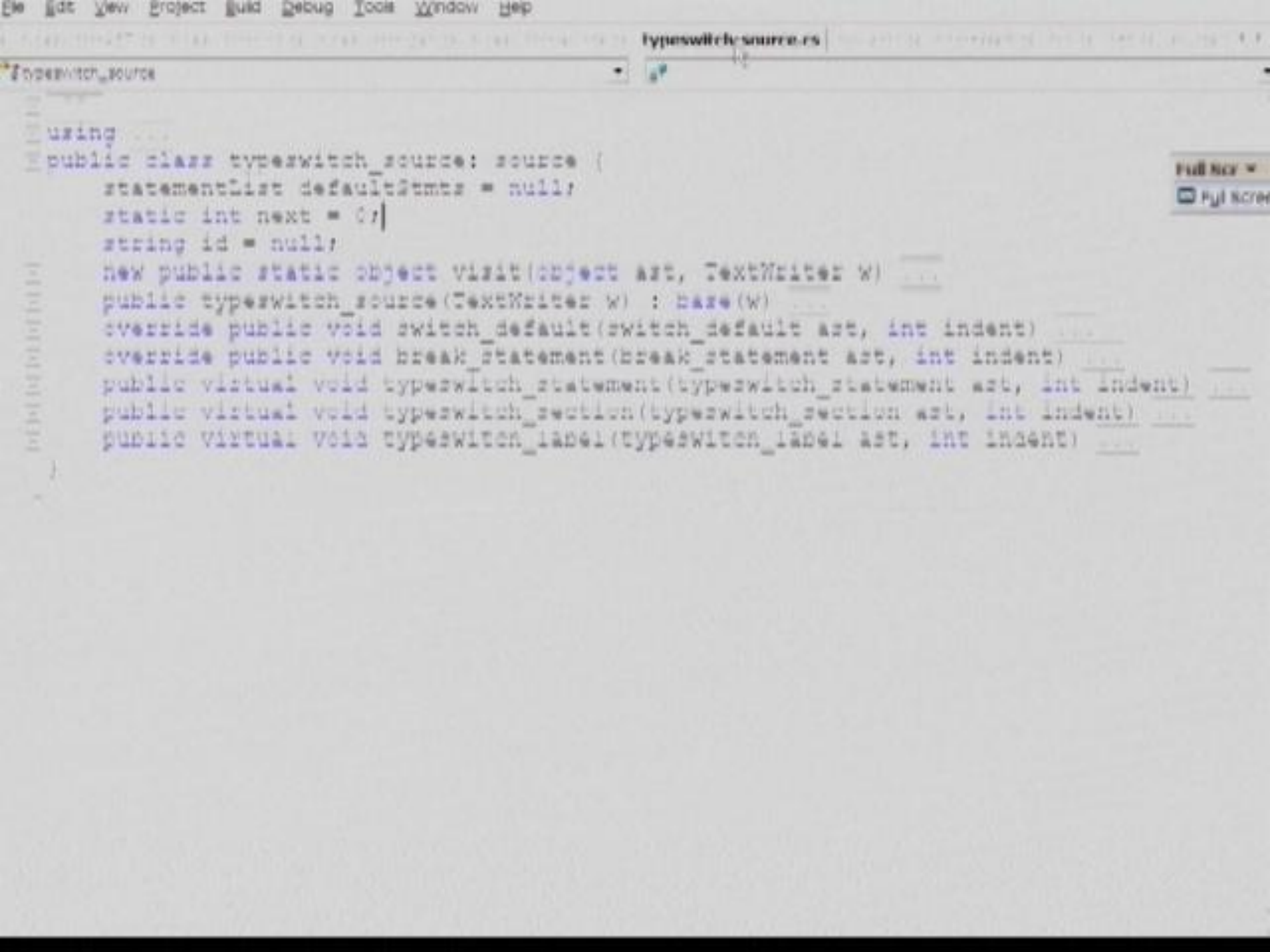
	A	B	C	
1	selection-statement	typeswitch-statement	statement	
2	typeswitch-statement	typeswitch (expression) typeswitch-block	statement	new typeswitch_statement(a3 a
3	typeswitch-block	{ typeswitch-sections$_{opt}$ }	IList	a2
4	typeswitch-sections	typeswitch-section	IList	typeswitch_sectionList.New(a1)
5	typeswitch-sections	typeswitch-sections typeswitch-section	IList	List.Cons(a1 a2)
6	typeswitch-section	case type (identifier) statement-list	typeswitch_section	new typeswitch_section(a2.a4.a
7	typeswitch-section	typeswitch-labels statement-list	typeswitch_section	new typeswitch_section(a1 a2)
8	typeswitch-section	default statement-list	typeswitch_section	new typeswitch_section(a3)
9	typeswitch-labels	typeswitch-label	IList	switch_labelList.New(a1)
10	typeswitch-labels	typeswitch-labels typeswitch-label	IList	List.Cons(a1 a2)
11	typeswitch-label	case type	switch_label	new typeswitch_label(a2)
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				

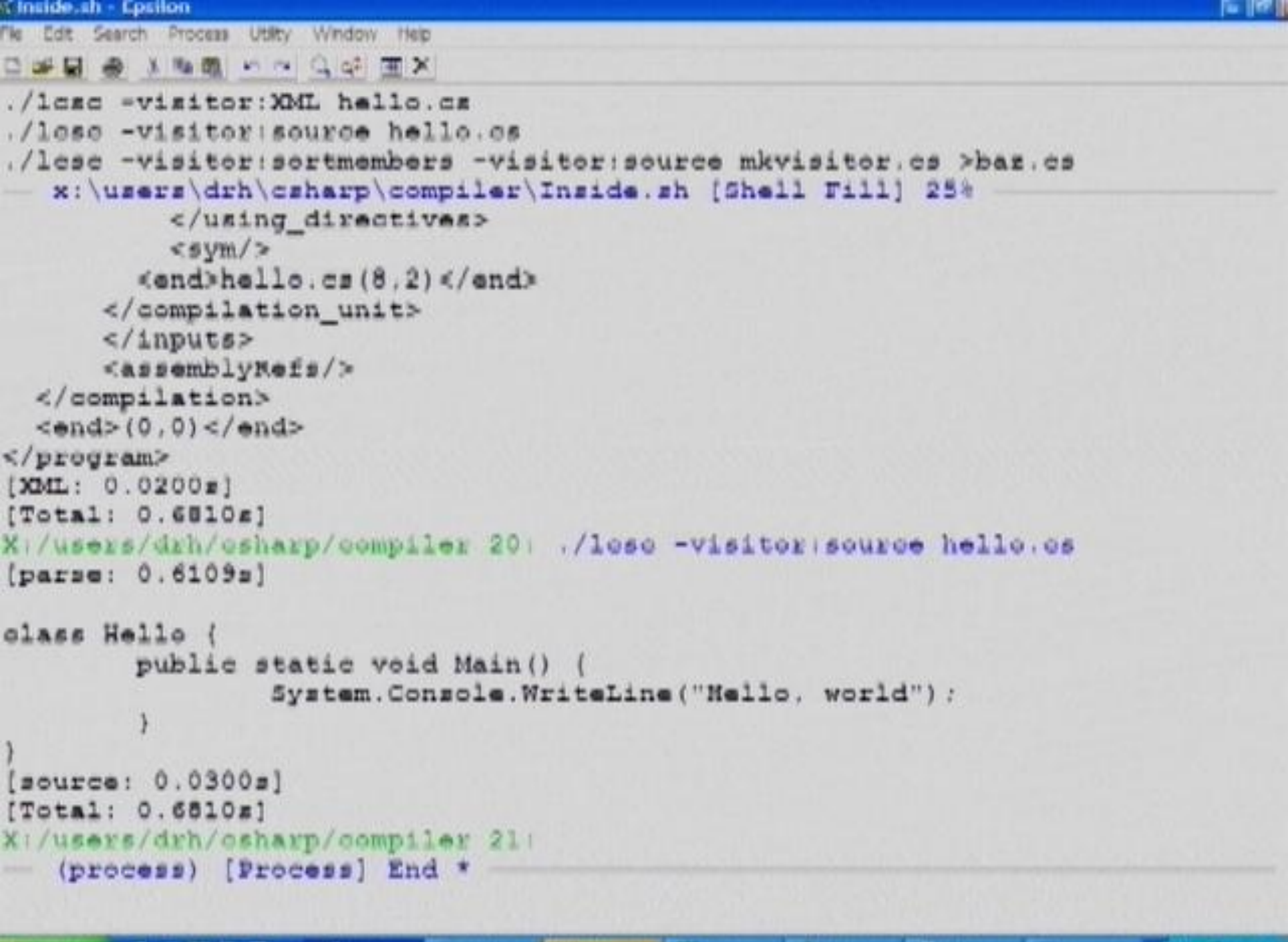

File Edit View Project Build Debug Tools Window Help
void fixed_parameter(fixed_parameter ast, SymbolTable bindings)
void fixed_statement(fixed_statement ast, SymbolTable bindings)
void float_type(float_type ast, SymbolTable bindings)
void for_decl(for_decl ast, SymbolTable bindings)
void for_init(for_init ast, SymbolTable bindings)
void for_list(for_list ast, SymbolTable bindings)
void for_statement(for_statement ast, SymbolTable bindings)
void foreach_statement(foreach_statement ast, SymbolTable bindings)
void formals(formals ast, SymbolTable bindings)
void goto_case_statement(goto_case_statement ast, SymbolTable bindings)
void goto_default_statement(goto_default_statement ast, SymbolTable bindings)
void goto_statement(goto_statement ast, SymbolTable bindings)
void if_statement(if_statement ast, symboltable bindings) {
if (ast.elsepart != null)
statement(ast.elsepart, bindings);
expression(ast.expr, bindings);
statement(ast.thenpart, bindings);
void implicit_cast_expression(implicit_cast_expression ast, symboltable bindings)
void implicit_declarator(implicit_declarator ast, symboltable bindings)
void indexer(indexer ast, symboltable bindings)
void indexer_declaration(indexer_declaration ast, symboltable bindings)
void int_type(int_type ast, symboltable bindings)
void integer_literal(integer_literal ast, symboltable bindings)
void interface_declaration(interface_declaration ast, symboltable bindings)
void interface_event_declaration(interface_event_declaration ast, SymbolTable bindi
void interface_indexer_declaration(interface_indexer_declaration ast, SymbolTable b
void interface_method_declaration(interface_method_declaration ast, SymbolTable bin

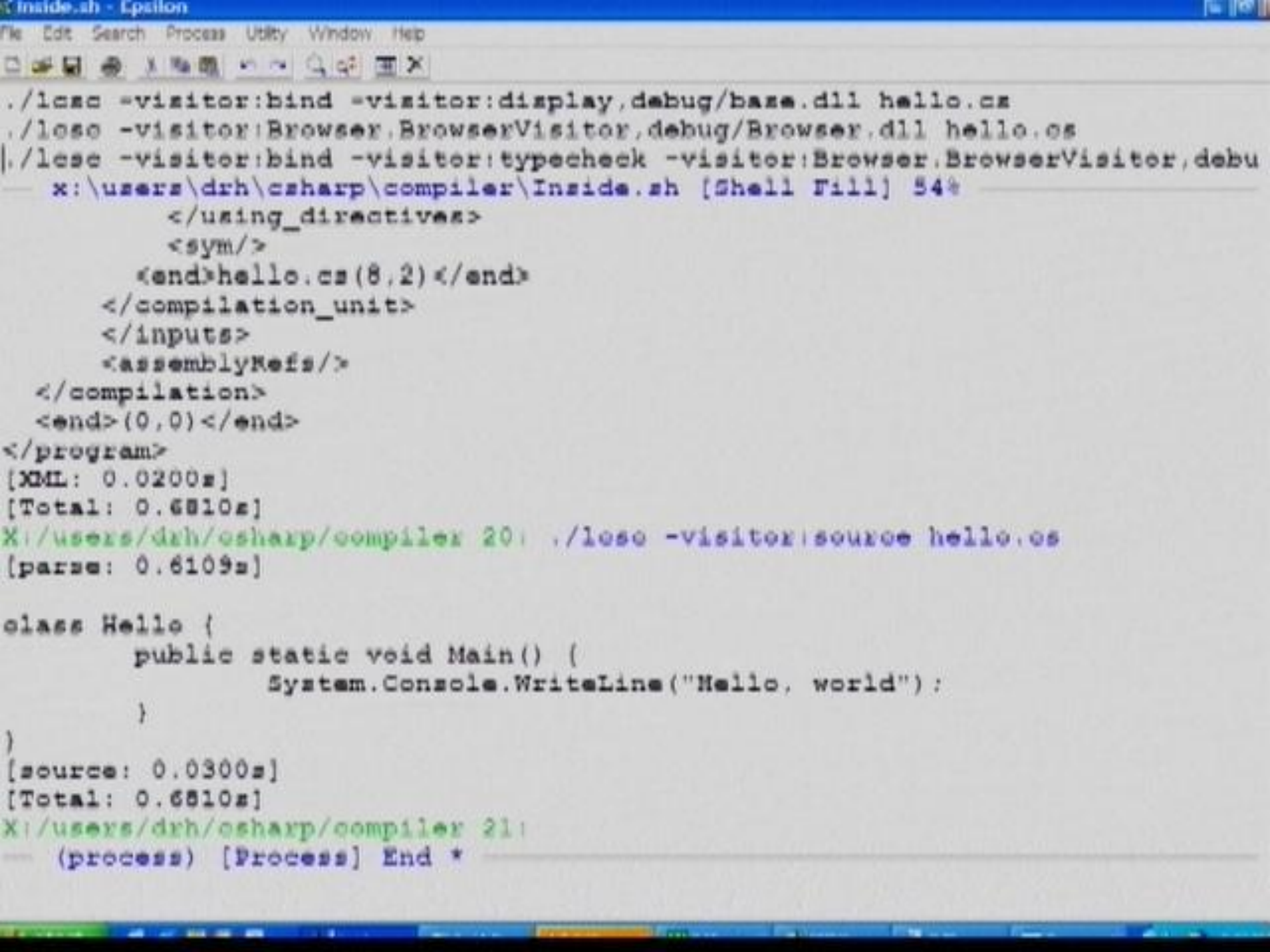
File Edit View Project Build Debug Tools Window Help

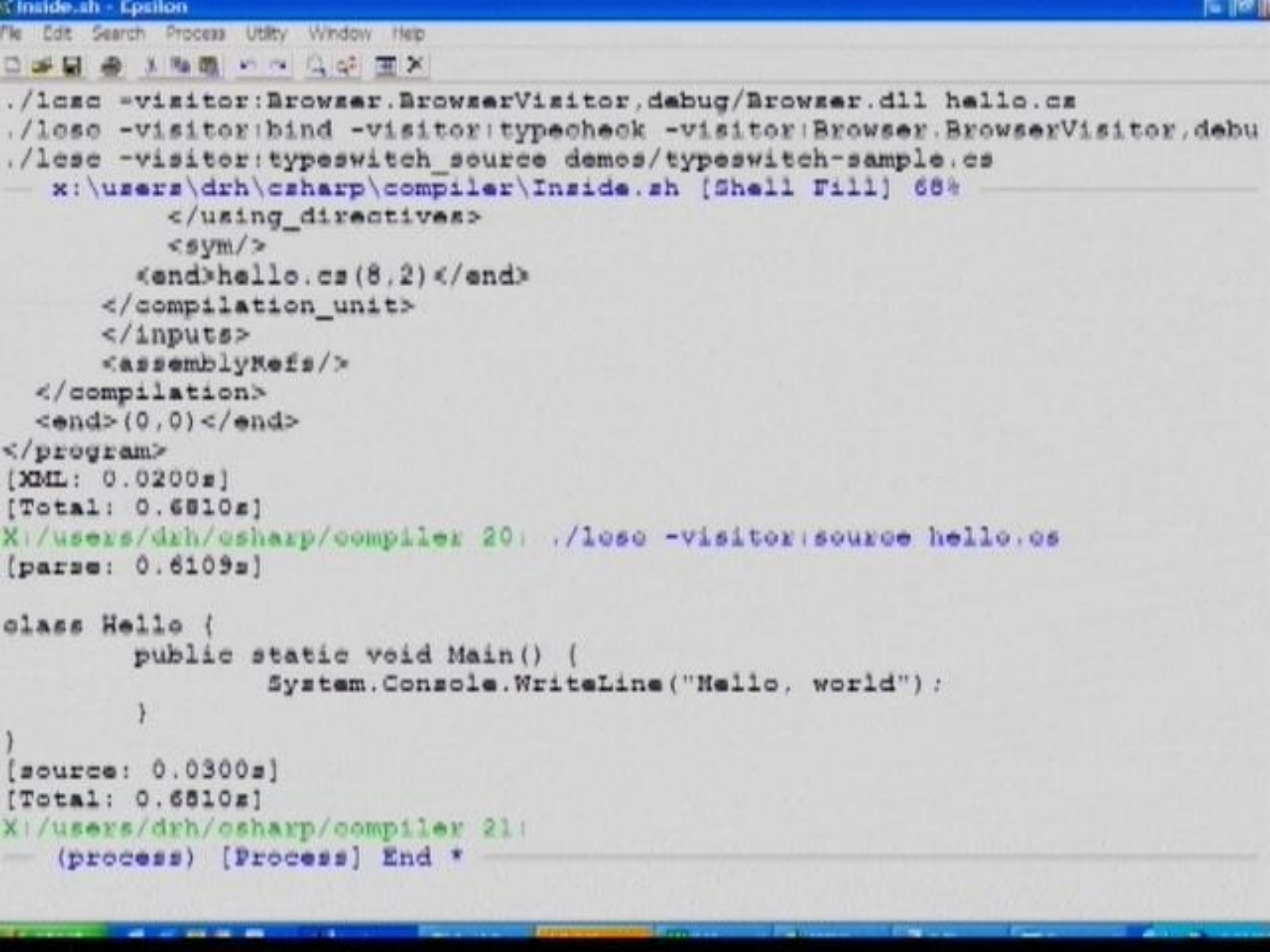
typeswitch-AST.cs

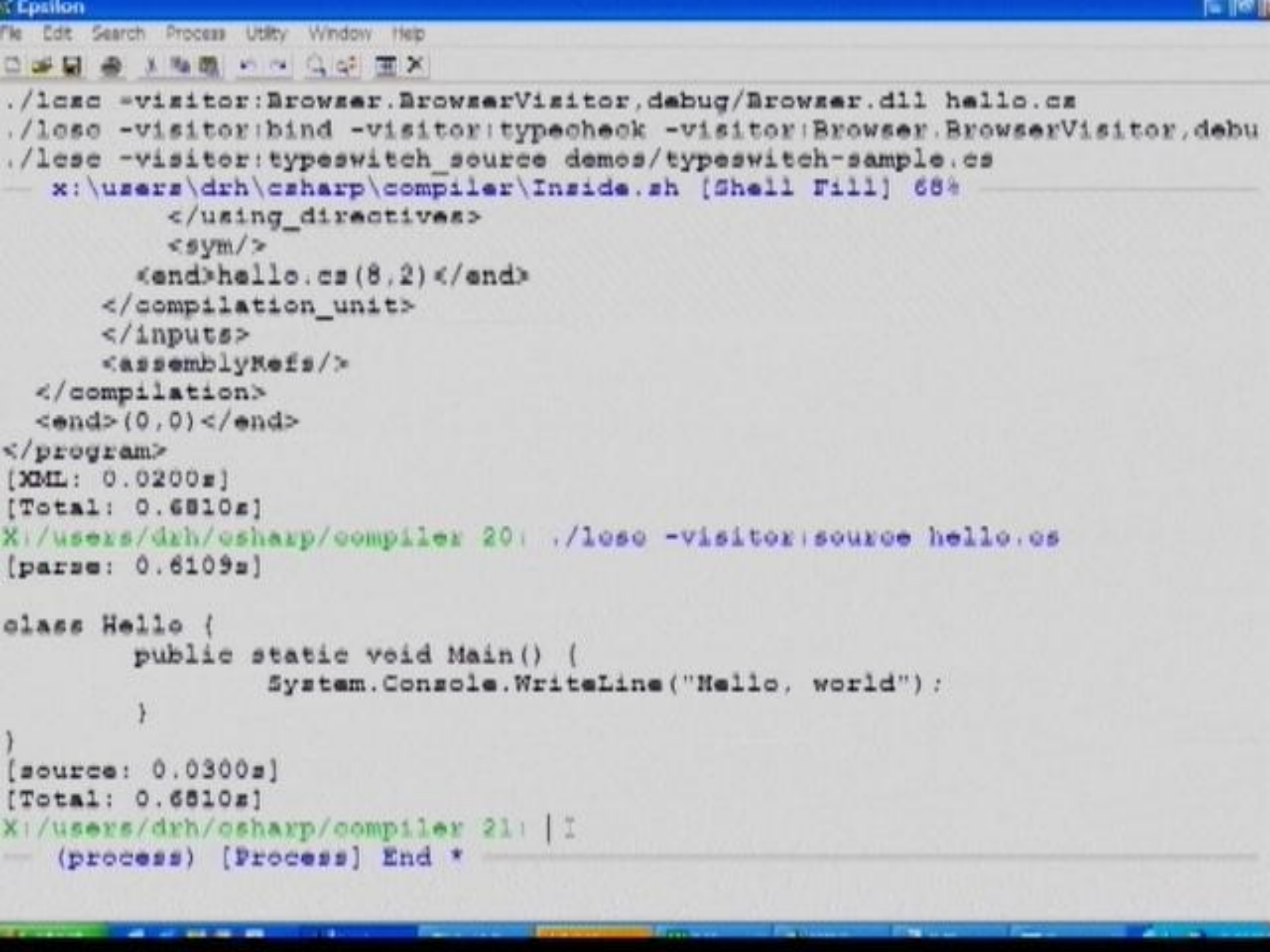
```
using ...
public class typeswitch_label: switch_label {
    public type typelabel;
    public typeswitch_label(type typelabel) ...
    public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
}

public class typeswitch_section: AST {
    [MaybeNull] public InputElement id;
    public switch_labelList labels;
    public statementList stmts;
    public typeswitch_section(IList labels, IList stmts) ...
    public typeswitch_section(type typelabel, InputElement id, IList stmts) ...
    public typeswitch_section(IList stmts) ...
    public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
    [bind1] public block block;          // bind1 block for local id
    [MaybeNull,bind2] public local sym; // bind2 symbol-table entry for id
}

public class typeswitch_statement: statement {
    public expression expr;
    public typeswitch_sectionList sections;
    public typeswitch_statement(expression expr, IList sections) ...
    public override void visit(ASTVisitor prefix, ASTVisitor postfix) ...
    public int var; // given temporary
}
```


File Edit View Project Build Debug Tools Window Help
typeswitch_source.cs
using
public class typeswitch_source: source {
statementlist defaultStmts = null;
static int next = 0;
string id = null;
new public static object visit(object ast, TextWriter w)
public typeswitch_source(TextWriter w) : base(w)
override public void switch_default(switch_default ast, int indent)
override public void break_statement(break_statement ast, int indent)
public virtual void typeswitch_statement(typeswitch_statement ast, int indent)
public virtual void typeswitch_section(typeswitch_section ast, int indent)
public virtual void typeswitch_label(typeswitch_label ast, int indent)
}


Inside.sh - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:XML hello.cs
./lcso -visitor:source hello.cs
./lcsc -visitor:sortmembers -visitor:source mkvisitor.cs >bar.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 25%
</using_directives>
<sym/>
<end>hello.cs(8,2)</end>
</compilation_unit>
</inputs>
<assemblyRefs/>
</compilation>
<end>(0,0)</end>
</program>
[XML: 0.0200s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 20: ./lcso -visitor:source hello.cs
[parse: 0.6109s]
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
[source: 0.0300s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 21:
(process) [Process] End *


Inside.sh - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:bind -visitor:display,debug/base.dll hello.cs
./lcso -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 54%
</using_directives>
<sym/>
<end>hello.cs(8,2)</end>
</compilation_unit>
</inputs>
<assemblyRefs/>
</compilation>
<end>(0,0)</end>
</program>
[XML: 0.0200s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 20: ./lcso -visitor:source hello.cs
[parse: 0.6109s]
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
[source: 0.0300s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 21:
(process) [Process] End *


Inside.sh - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./losc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lesc -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
</using_directives>
<sym/>
<end>hello.cs(8,2)</end>
</compilation_unit>
</inputs>
<assemblyRefs/>
</compilation>
<end>(0,0)</end>
</program>
[XML: 0.0200s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 20: ./losc -visitor:source hello.cs
[parse: 0.6109s]
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
[source: 0.0300s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 21:
(process) [Process] End *


Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcso -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcse -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
</using_directives>
<sym/>
<end>hello.cs(8,2)</end>
</compilation_unit>
</inputs>
<assemblyRefs/>
</compilation>
<end>(0,0)</end>
</program>
[XML: 0.0200s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 20: ./lcso -visitor:source hello.cs
[parse: 0.6109s]
class Hello {
public static void Main() {
System.Console.WriteLine("Hello, world");
}
}
[source: 0.0300s]
[Total: 0.6810s]
X:/users/drh/csharp/compiler 21:
(process) [Process] End *

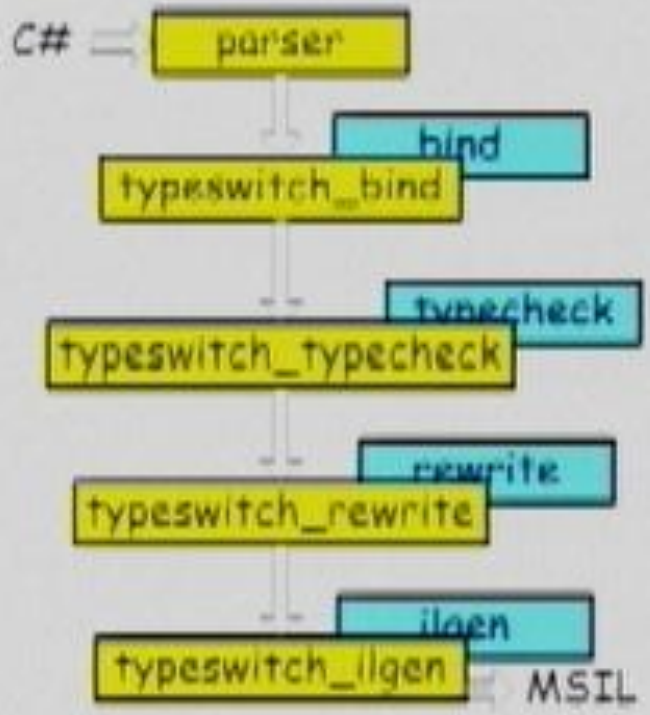
```
Epsilon
File  Edit  Search  Process  Utility  Window  Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
   x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
                        if (yy_1 is Int32) {
                                Int32 x = (Int32)yy_1;
                                Console.WriteLine(x);
                                goto yy_1_end;
                        }
                        if (yy_1 is Symbol) {
                                Symbol s = (Symbol)yy_1;
                                Symbols.Add(s);
                                goto yy_1_end;
                        }
                        if (yy_1 is Segment) {
                                popSegment();
                                goto yy_1_end;
                        }
                        throw new ArgumentException();
                yy_1_end: ;
                        }
        }
}
[typeswitch_source: 0.0501s]
[Total: 1.3219s]
X:/users/drh/csharp/compiler 22:
   (process) [Process] End *
```

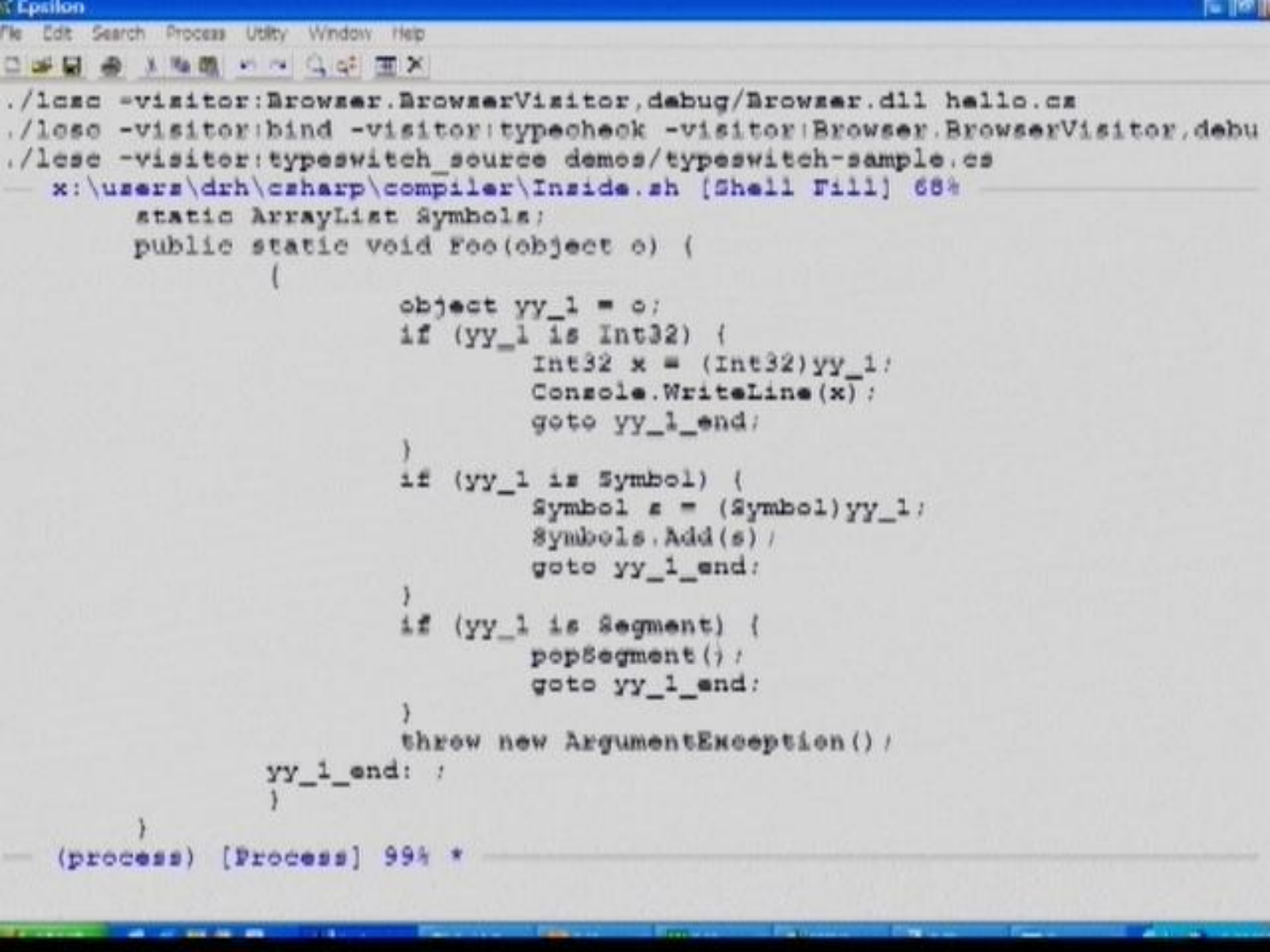
The Easy Way: Source-to-Source Transformation

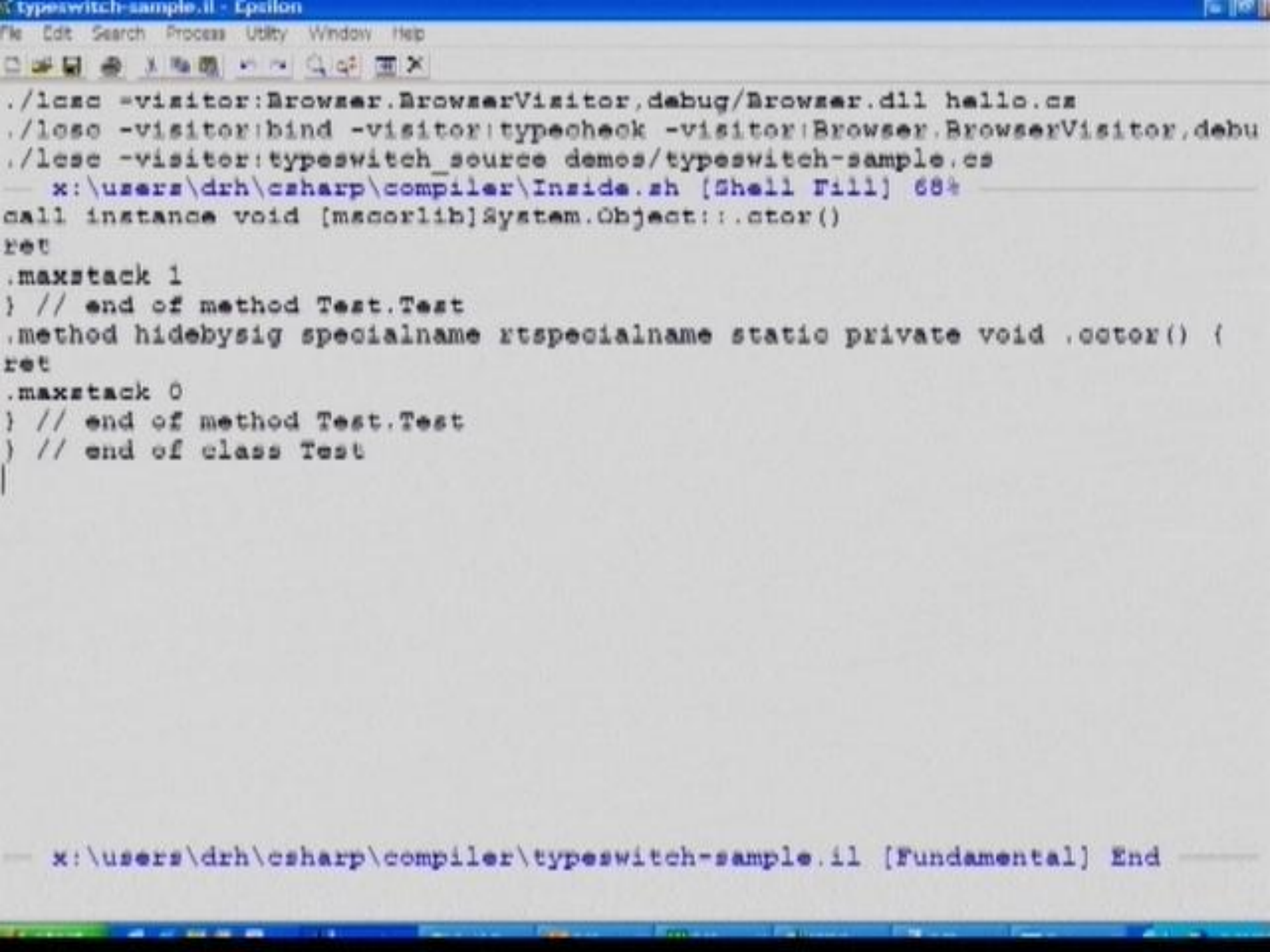
- Prepare grammar spreadsheet (1+11+7=19 lines) [Demo]
- Add typeswitch AST nodes (63 lines) [Demo]
- *Extend* source.cs (71 lines) [Demo]
- Alternative
  - Build appropriate if-else ASTs directly, or rewrite AST
  - Use source.cs unmodified

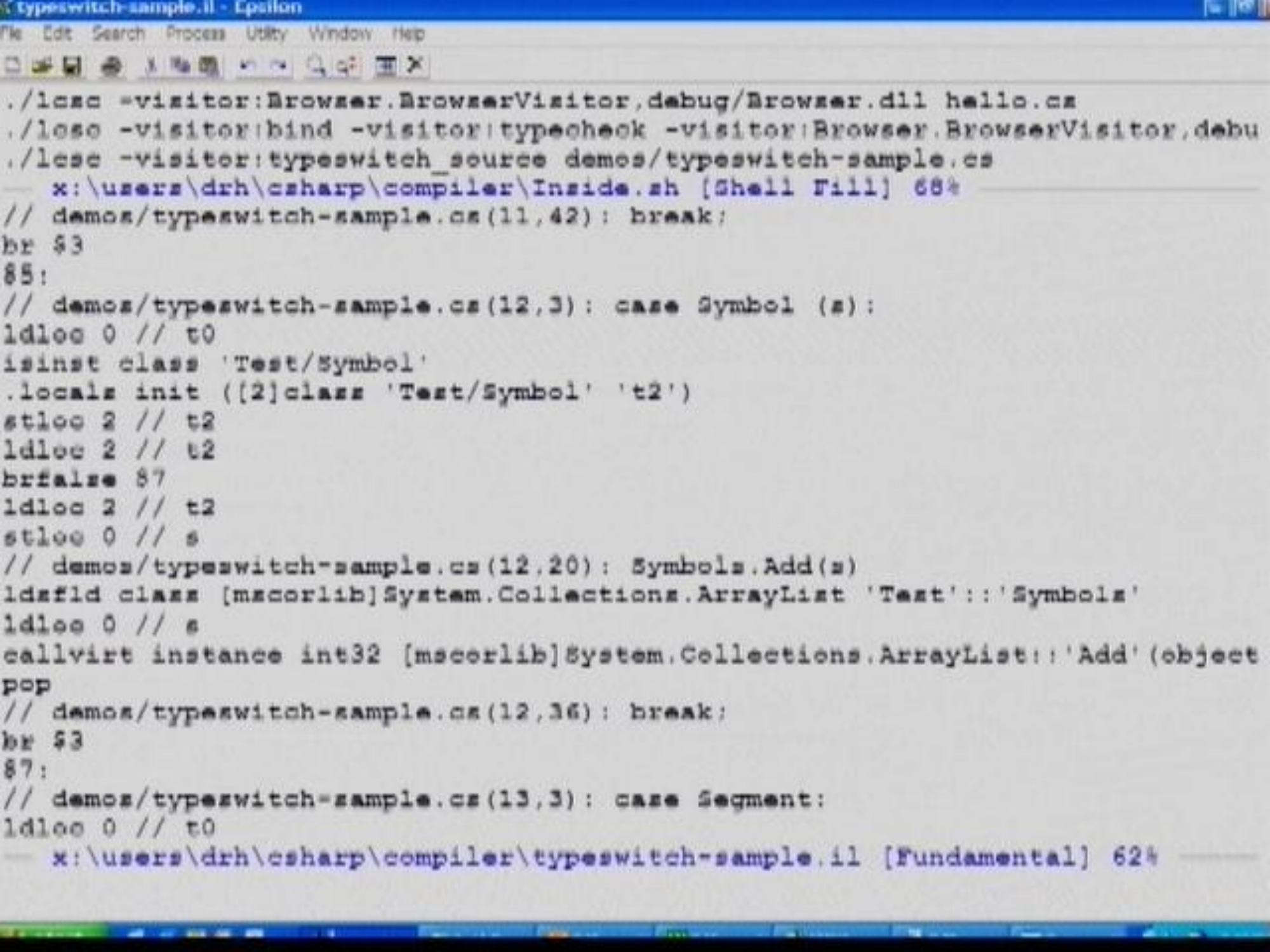
The Full 9 Yards: Emitting MSIL

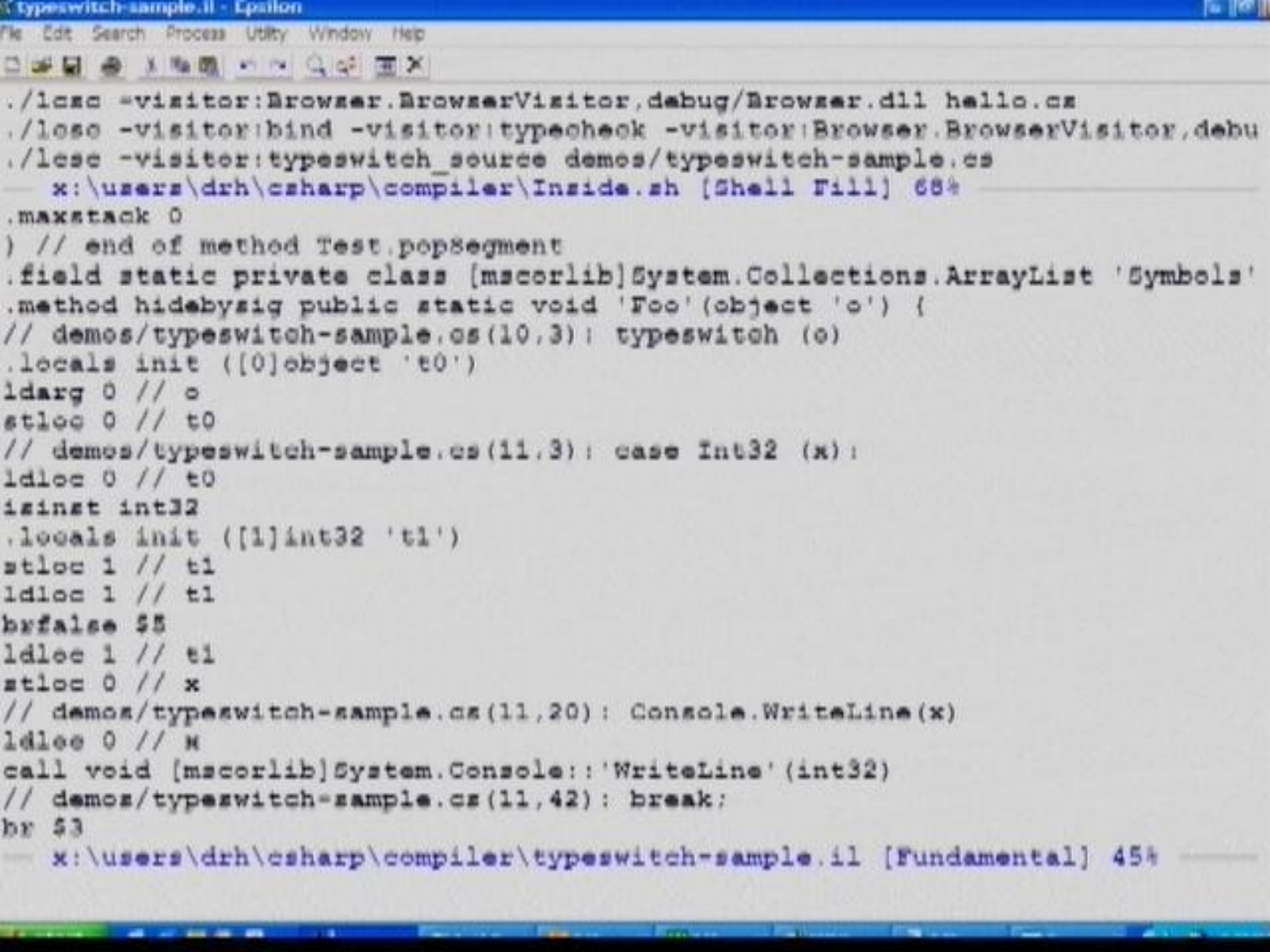
- Extend bind.cs (100 lines)
- Extend typecheck.cs (35 lines)
- Extend rewrite.cs (29 lines)
- Extend ilgen.cs (65 lines)
- [Demo]
- Alternative
  - Build appropriate if-else ASTs directly, or rewrite AST as first visitor
  - Use bind.cs, typecheck.cs, rewrite.cs, ilgen.cs unmodified

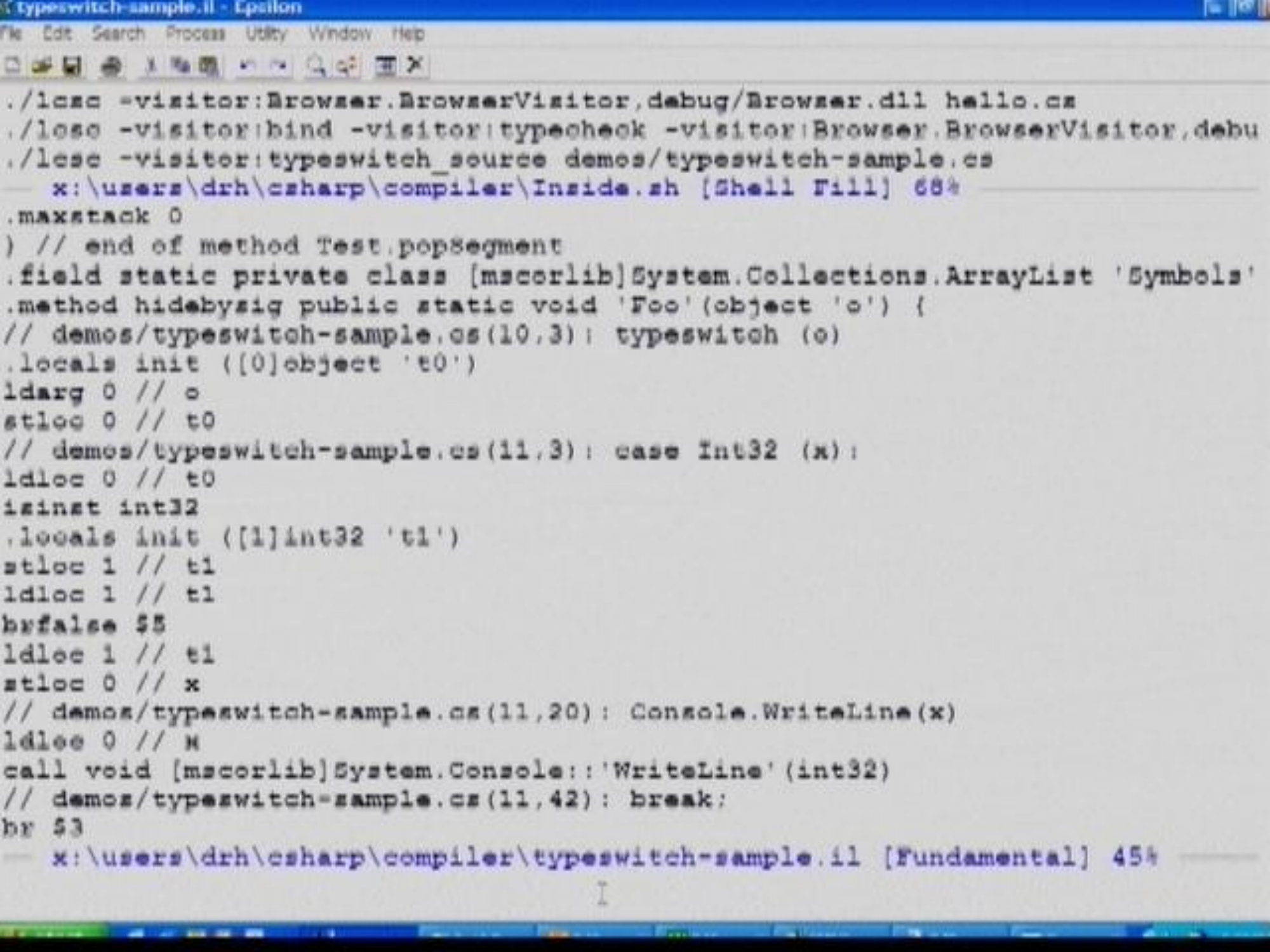



Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
static ArrayList Symbols;
public static void Foo(object o) {
{
object yy_1 = o;
if (yy_1 is Int32) {
Int32 x = (Int32)yy_1;
Console.WriteLine(x);
goto yy_1_end;
}
if (yy_1 is Symbol) {
Symbol s = (Symbol)yy_1;
Symbols.Add(s);
goto yy_1_end;
}
if (yy_1 is Segment) {
popSegment();
goto yy_1_end;
}
throw new ArgumentException();
yy_1_end: ;
}
}
(process) [Process] 99% *


typeswitch-sample.il - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
call instance void [mscorlib]System.Object::.ctor()
ret
.maxstack 1
} // end of method Test.Test
.method hidebysig specialname rtspecialname static private void .cctor() {
ret
.maxstack 0
} // end of method Test.Test
} // end of class Test
x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] End


typeswitch-sample.il - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcso -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
// demos/typeswitch-sample.cs(11,42): break;
br $3
$5:
// demos/typeswitch-sample.cs(12,3): case Symbol (s):
ldloc 0 // t0
isinst class 'Test/Symbol'
.locals init ([2]class 'Test/Symbol' 't2')
stloc 2 // t2
ldloc 2 // t2
brfalse $7
ldloc 2 // t2
stloc 0 // s
// demos/typeswitch-sample.cs(12,20): Symbols.Add(s)
ldsfld class [mscorlib]System.Collections.ArrayList 'Test'::'Symbols'
ldloc 0 // s
callvirt instance int32 [mscorlib]System.Collections.ArrayList::'Add'(object
pop
// demos/typeswitch-sample.cs(12,36): break;
br $3
$7:
// demos/typeswitch-sample.cs(13,3): case Segment:
ldloc 0 // t0
x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] 62%


typeswitch-sample.il - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
.maxstack 0
} // end of method Test.popSegment
.field static private class [mscorlib]System.Collections.ArrayList 'Symbols'
.method hidebysig public static void 'Foo'(object 'o') {
// demos/typeswitch-sample.cs(10,3): typeswitch (o)
.locals init ([0]object 't0')
ldarg 0 // o
stloc 0 // t0
// demos/typeswitch-sample.cs(11,3): case Int32 (x):
ldloc 0 // t0
isinst int32
.locals init ([1]int32 't1')
stloc 1 // t1
ldloc 1 // t1
brfalse $5
ldloc 1 // t1
stloc 0 // x
// demos/typeswitch-sample.cs(11,20): Console.WriteLine(x)
ldloc 0 // x
call void [mscorlib]System.Console::'WriteLine'(int32)
// demos/typeswitch-sample.cs(11,42): break;
br $3
x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] 45%


typeswitch-sample.il - Epsilon
File Edit Search Process Utility Window Help
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./losc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lesc -visitor:typeswitch_source demos/typeswitch-sample.cs
x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
.maxstack 0
} // end of method Test::popSegment
.field static private class [mscorlib]System.Collections.ArrayList 'Symbols'
.method hidebysig public static void 'Foo'(object 'o') {
// demos/typeswitch-sample.cs(10,3): typeswitch (o)
.locals init ([0]object 't0')
ldarg 0 // o
stloc 0 // t0
// demos/typeswitch-sample.cs(11,3): case Int32 (x):
ldloc 0 // t0
isinst int32
.locals init ([1]int32 't1')
stloc 1 // t1
ldloc 1 // t1
brfalse $5
ldloc 1 // t1
stloc 0 // x
// demos/typeswitch-sample.cs(11,20): Console.WriteLine(x)
ldloc 0 // x
call void [mscorlib]System.Console::'WriteLine'(int32)
// demos/typeswitch-sample.cs(11,42): break;
br $3
x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] 45%

typeswitch-sample.il - Epsilon

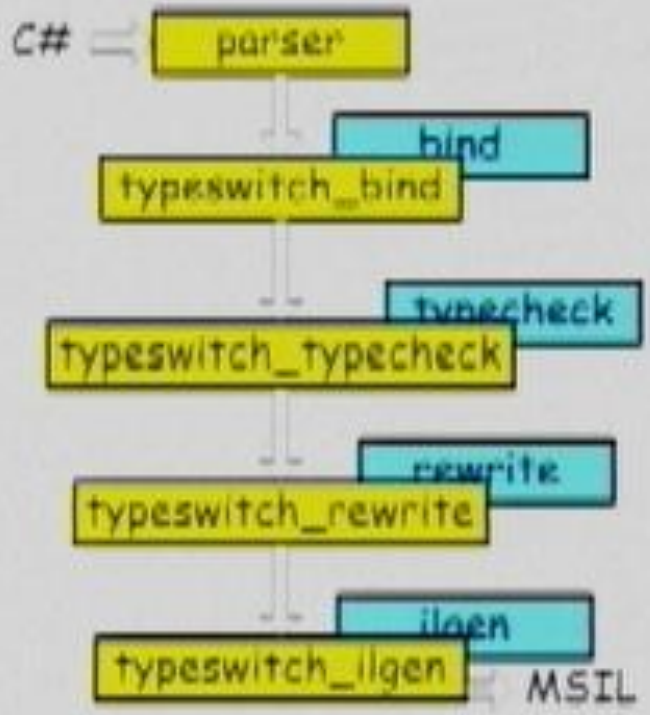
File Edit Search Process Utility Window Help

```
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
   x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 68%
.maxstack 0
) // end of method Test.popSegment
.field static private class [mscorlib]System.Collections.ArrayList 'Symbols'
.method hidebysig public static void 'Foo'(object 'o') {
// demos/typeswitch-sample.cs(10,3): typeswitch (o)
.locals init ([0]object 't
ldarg 0 // o
stloc 0 // t0
// demos/typeswitch-sample
ldloc 0 // t0
isinst int32
.locals init ([1]int32 't1
stloc 1 // t1
ldloc 1 // t1
brfalse $5
ldloc 1 // t1
stloc 0 // x
// demos/typeswitch-sample.cs(11,20): Console.WriteLine(x)
ldloc 0 // x
call void [mscorlib]System.Console::'WriteLine'(int32)
// demos/typeswitch-sample.cs(11,42): break;
br $3
   x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] 44%
```

PowerPoint Slide Show - [Inside.ppt]

The Full 9 Yards: Emitting MSIL

- Extend bind.cs (100 lines)
- Extend typecheck.cs (35 lines)
- Extend rewrite.cs (29 lines)
- Extend ilgen.cs (65 lines)
- [Demo]
- Alternative
  - Build appropriate if-else ASTs directly, or rewrite AST as first visitor
  - Use bind.cs, typecheck.cs, rewrite.cs, ilgen.cs unmodified



Recap

- A simple C# compiler
  - Easy to understand (at least relatively)
  - Easy to modify
  - Ideal for language design experiments, language-based tools
- More info
  - http://msrweb/pls/csharp/compiler/
  - Source Depot port MSR:5010, //pls/csharp/compiler/...
  - Current snapshot in \\research\Root\Public\drh\csharp\compiler